# Machine Learning For Design

## Lecture 8 - Designing And Develop Machine Learning Models / Part 2

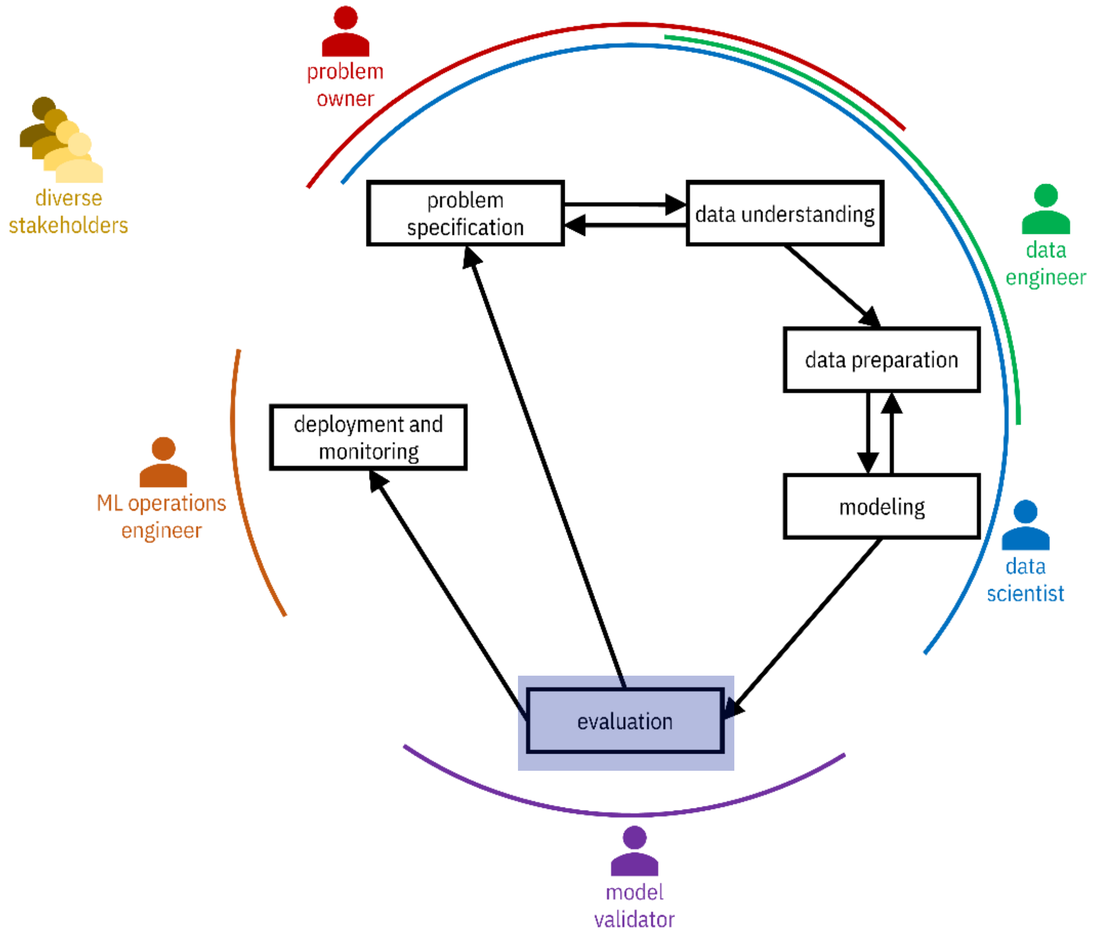
Alessandro Bozzon
23/03/2022

mlfd-io@tudelft.nl
www.ml4design.com

# Evaluation

# Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology

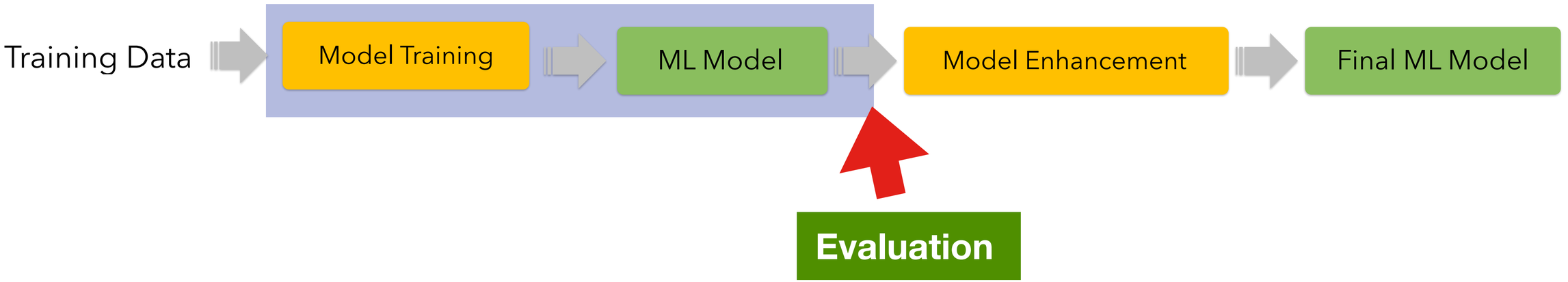
**Lecture 2**



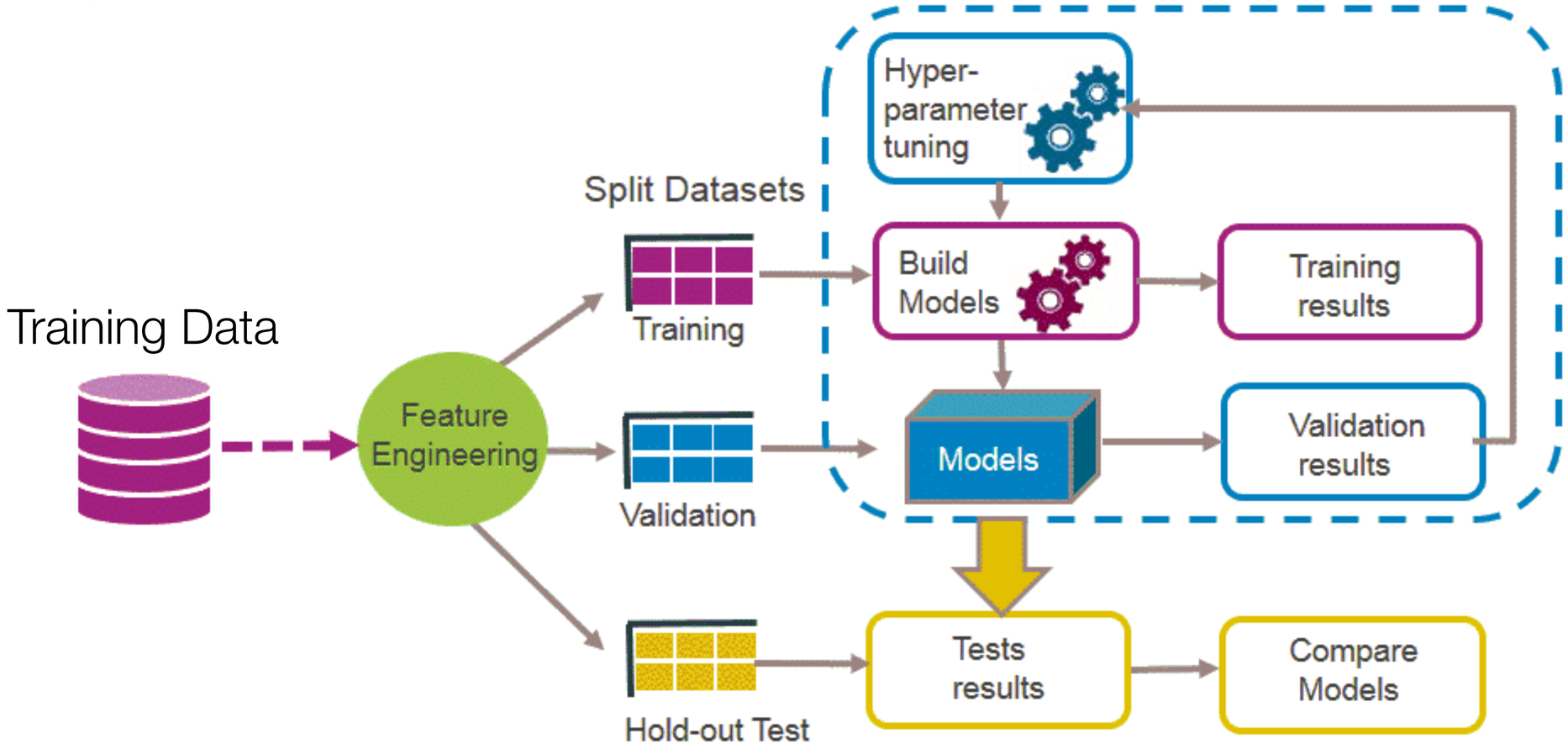

https://www.the-modeling-agency.com/crisp-dm.pdf

# How do machines learn?

**Lecture 2**

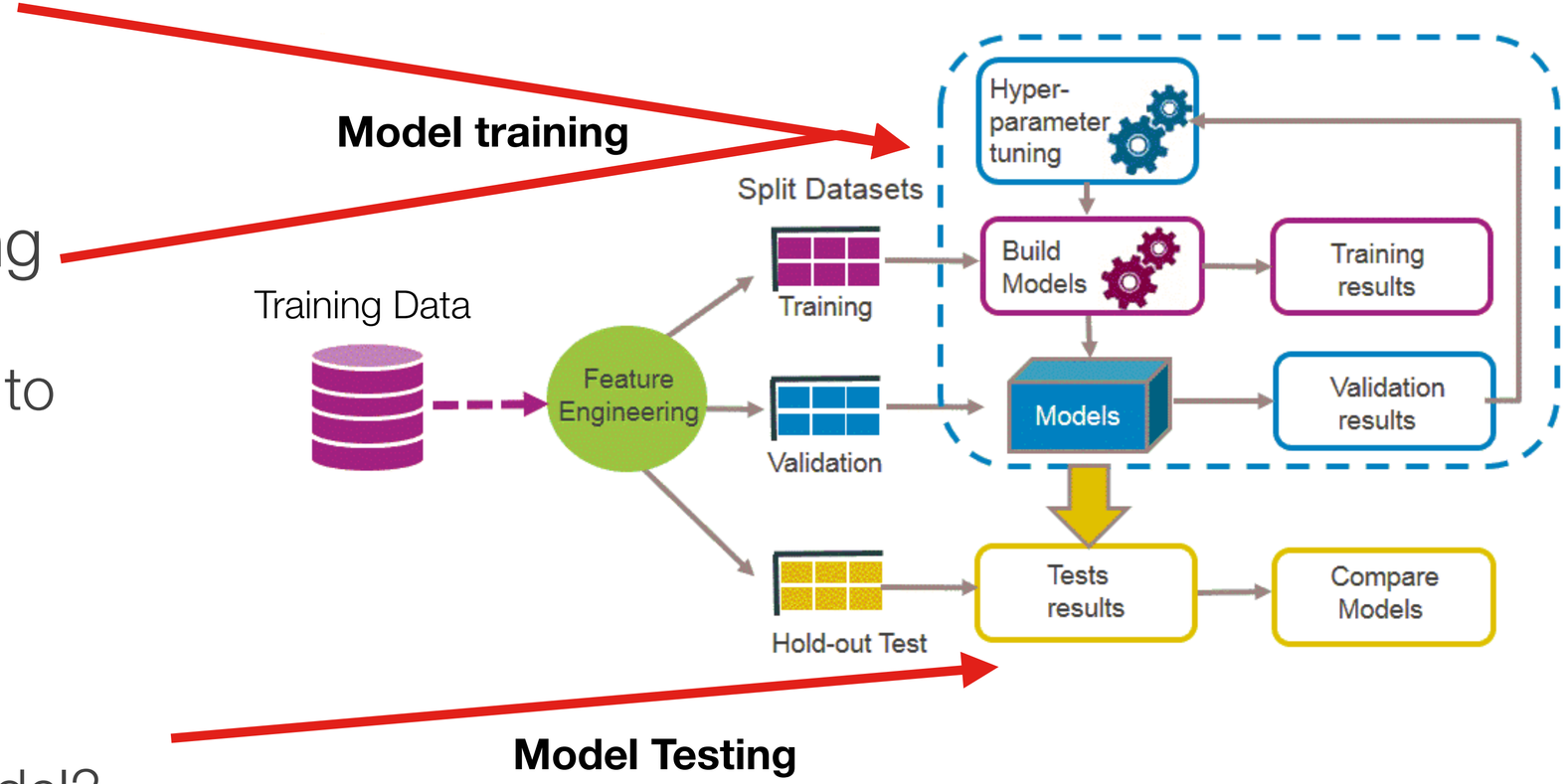


# Machine Learning Training and Evaluation Process



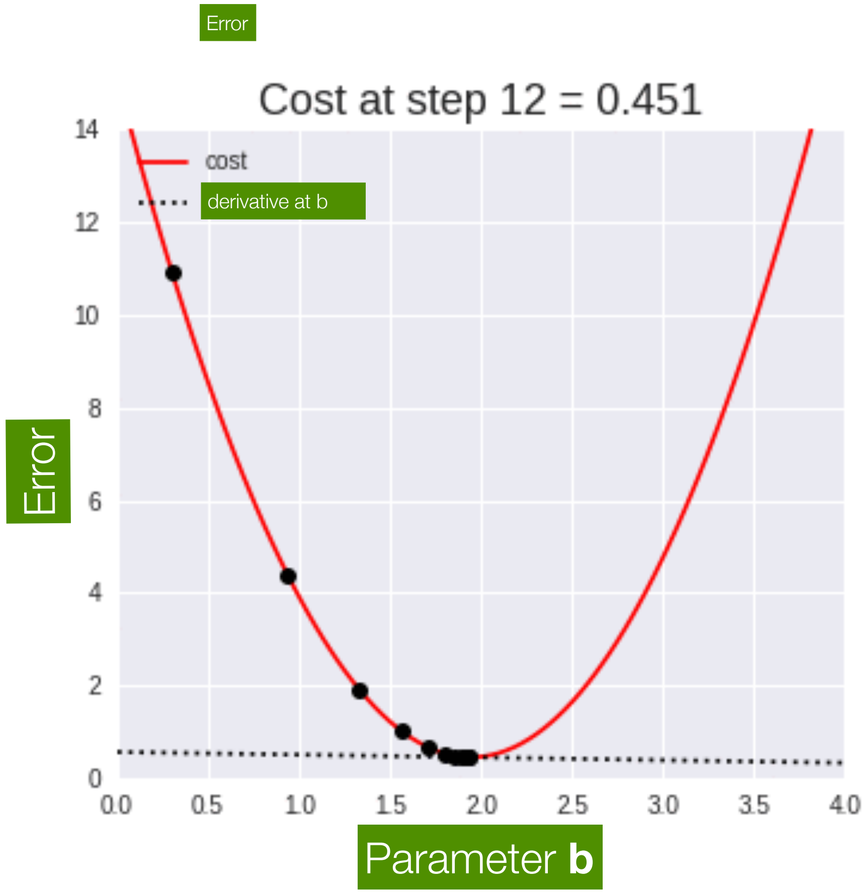
# How to Evaluate?

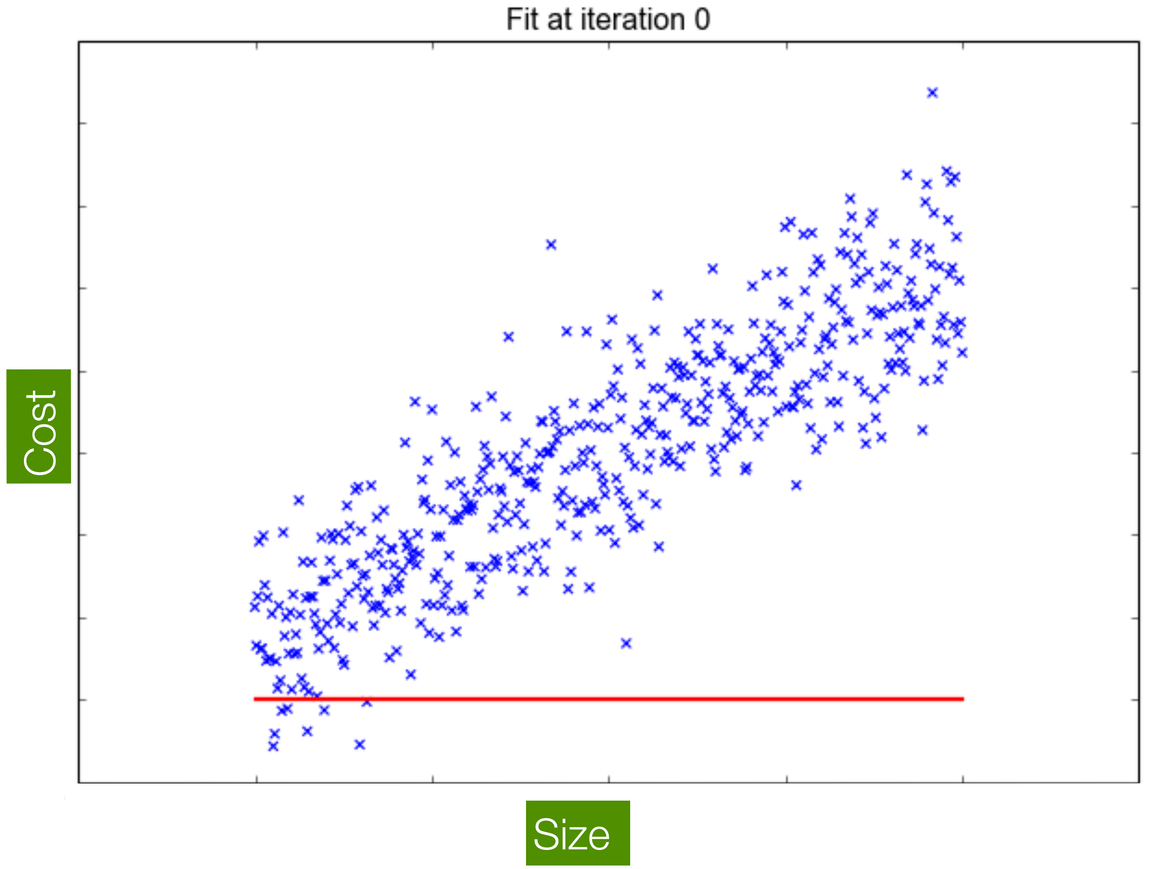
- Metric
  - How to measure errors?
  - Both training and testing
- Training of Machine Learning algorithm
  - How to “help” the ML model to generalise?
- Experiment
  - How to pick the best ML model?



# Training the model

## Gradient descent



# Model Training: Metric

- Errors are almost inevitable!
  - How to measure errors?

- We're generally interested in the following:
  - How often is the prediction wrong?
  - How is the prediction wrong?
  - What is the cost of wrong predictions?
  - How does the cost vary by the type of prediction that was wrong?
  - How can we minimize costs? (or regret?)

- Select an evaluation procedure (a "metric")
  - **Ok, but which one?**

# Regression

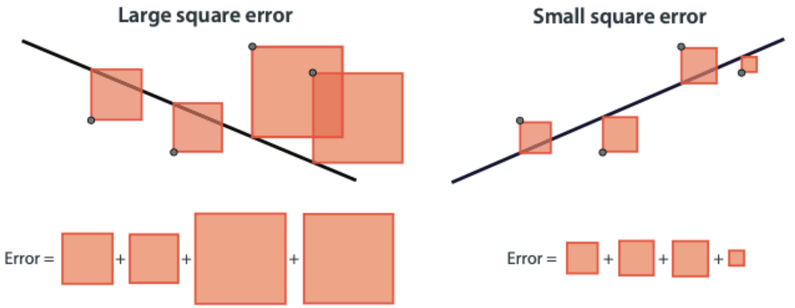
- **Mean (absolute | square) error**

- **Absolute**: average of the difference between the original values and the predicted value
  - No direction

$$MAE = \frac{1}{N}\sum_{J=1}^{N} |prediction_j - value_j|$$


- **Square**: average of the square of the difference between the original values and the predicted value
  - Square is *nicer* to deal with during the training process (derivative)
  - Larger errors are more pronounced

$$MSE = \frac{1}{2N}\sum_{J=1}^{N} (prediction_j - value_j)^2$$

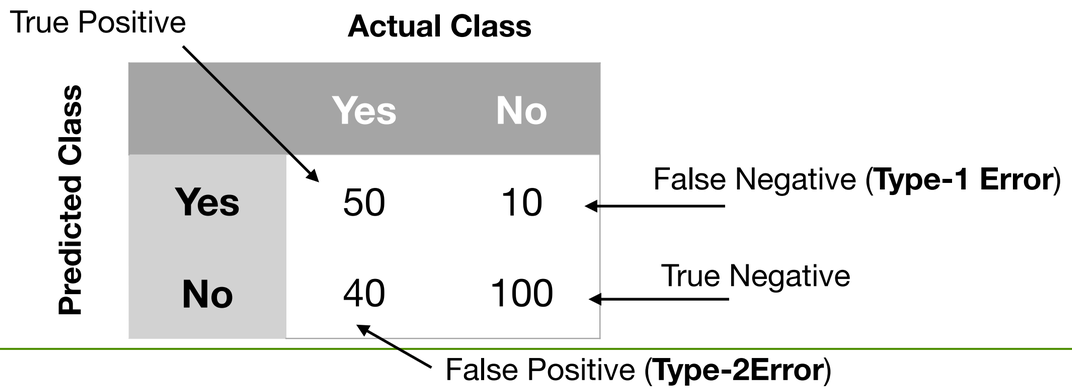


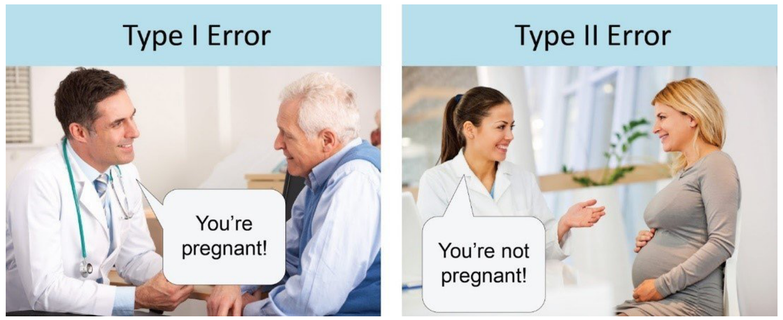
# Classification

- **Accuracy**
  - The percentage of times that a model is correct
  - However,  the model with the highest accuracy is not necessarily the best model
  - Some errors (e.g. False Negative) may be much more expensive than others
    - Usually due to imbalanced trained datasets

$$Accuracy = \frac{\#CorrectPredictions}{\#Predictions}$$

- **Confusions Matrix**
  - Describes the complete performance of the model




$$Accuracy = \frac{\#TruePositives + \#TrueNegatives}{\#AllPredictions}$$

# All errors are not equal

- Depending on your task, different errors have different costs

- Pregnancy detection
  - Cost of “false negatives”?
  - Cost of “false positives”?

- Covid testing
  - Cost of “false negatives”?
  - Cost of “false positives”?

- In law enforcement?

- In detecting the “Alexa” command?
- In detecting a person on the road?

## FALSE POSITIVES: SELF-DRIVING CARS AND THE AGONY OF KNOWING WHAT MATTERS


According to a preliminary report released by the National Transportation Safety Board last week, Uber’s system detected pedestrian Elaine Herzberg six seconds before striking and killing her. It identified her as an unknown object, then a vehicle, then finally a bicycle. (She was pushing a bike, so close enough.) About a second before the crash, the system determined it needed to slam on the brakes. But Uber hadn’t set up its system to act on that decision, the NTSB explained in the report. The engineers prevented their car from making that call on its own “to reduce the potential for erratic vehicle behavior.” (The company relied on the car’s human operator to avoid crashes, which is a whole separate problem.)

ARN MORE

Uber’s engineers decided not to let the car auto-brake because they were worried the system would overreact to things that were unimportant or not there at all. They were, in other words, very worried about false positives.

https://www.wired.com/story/self-driving-cars-uber-crash-false-positive-negative/

# Classification

- **Precision**
  - Among the positive examples, how many did we correctly classify?

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

- **Recall**
  - Among the examples we classified as positive, how many did we correctly classify?

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

- **F1-Score**
  - The harmonic mean between **precision** (how many instances correctly classified), and **recall** (how many relevant instances are correctly classified)

$$F_1 = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}}$$

  - What is the implicit assumption about the costs of errors?

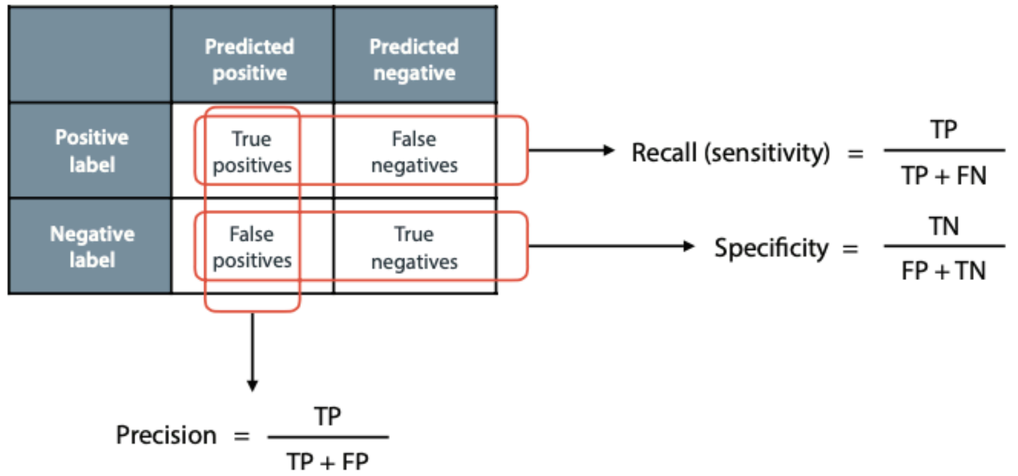
# Classification

- **Sensitivity** (True positive rate)
  - the capacity of the model to identify the positively labelled points
  - Same as recall

$$Sensitivity = \frac{TruePositive}{FalseNegative + TruePositive}$$

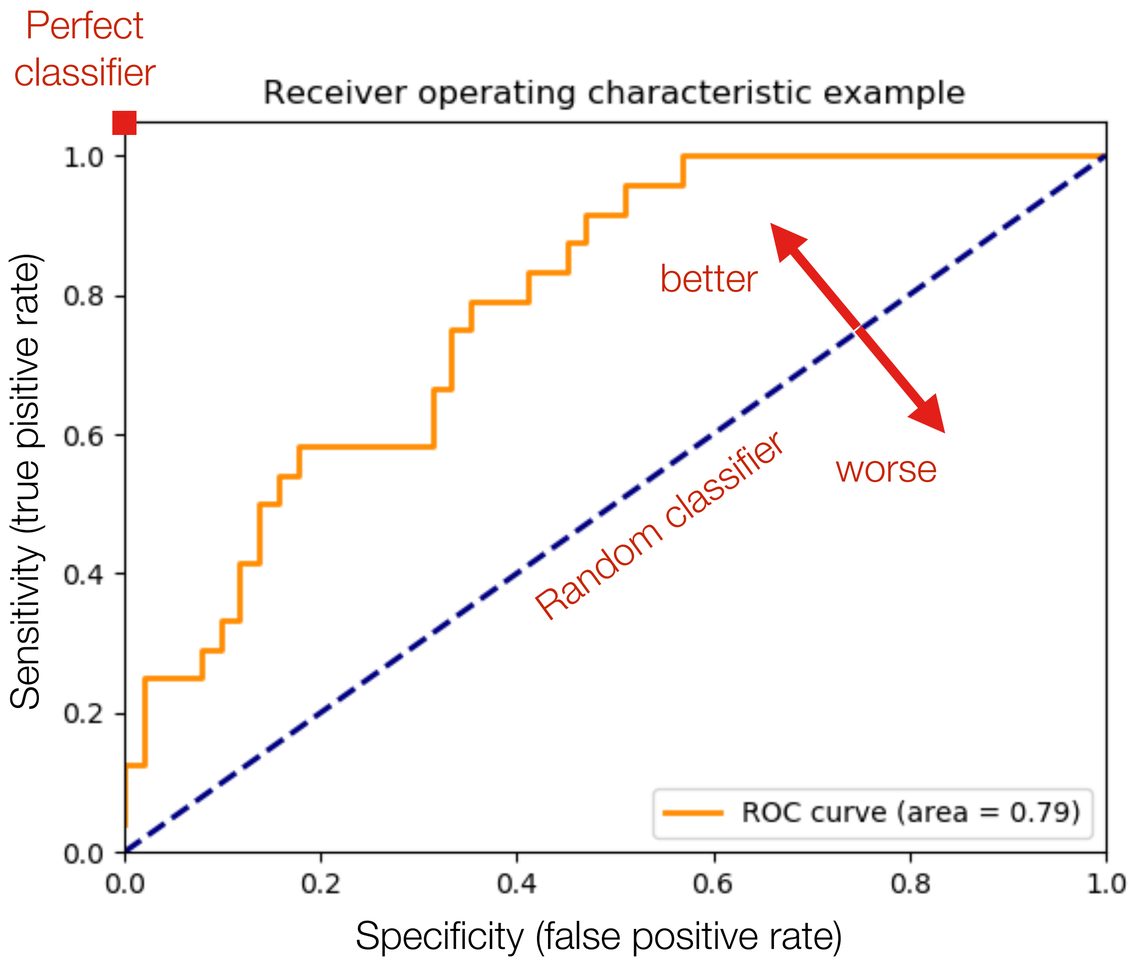
- **Specificity** (False positive rate)
  - the capacity of the model to identify the negatively labeled points
  - Not the same as precision

$$Specificity = \frac{FalsePositive}{FalsePositive + TrueNegative}$$



# The receiver operating characteristic (ROC) curve

- A useful technique to evaluate a model based on its performance on false positives and negatives at the same time
  - based on *sensitivity* and *specificity*
- It also gives us a way to "explore" model performance visually
  - Trade-off specificity and sensitivity by moving the threshold



# Some Examples

- Medical model:
  - **Recall** and **sensitivity**: among the sick people (positives), how many were correctly diagnosed as sick?
  - **Precision**: among the people diagnosed as sick, how many were actually sick?
  - **Specificity**: among the healthy people (negatives), how many were correctly diagnosed as healthy?
- Email model:
  - **Recall** and **sensitivity**: among the spam emails (positives), how many were correctly deleted?
  - **Precision**: among the deleted emails, how many were actually spam?
  - **Specificity**: among the ham emails (negatives), how many were correctly sent to the inbox?

# Choosing Metrics

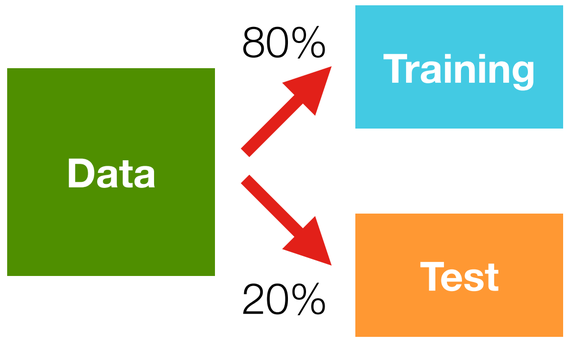
- If a high precision is a hard constraint, do the best recall
  - search engine results, grammar correction: Intolerant to FP
  - Metric: Recall at Precision = XX %

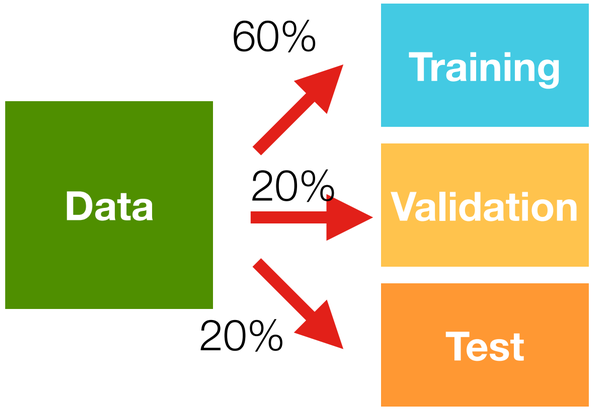
- If a high recall is a hard constraint, do best precision
  - medical diagnosis: Intolerant to FN
  - Metric: Precision at Recall = 100 %

- Capacity constrained (by K)
  - Metric: Precision in top-K

# Model Training: dataset splitting

- Split your data
  - Training set —> to **train** the model
  - (Optional) Validation set —> to decide which model to use
  - Test set —> to evaluate the model

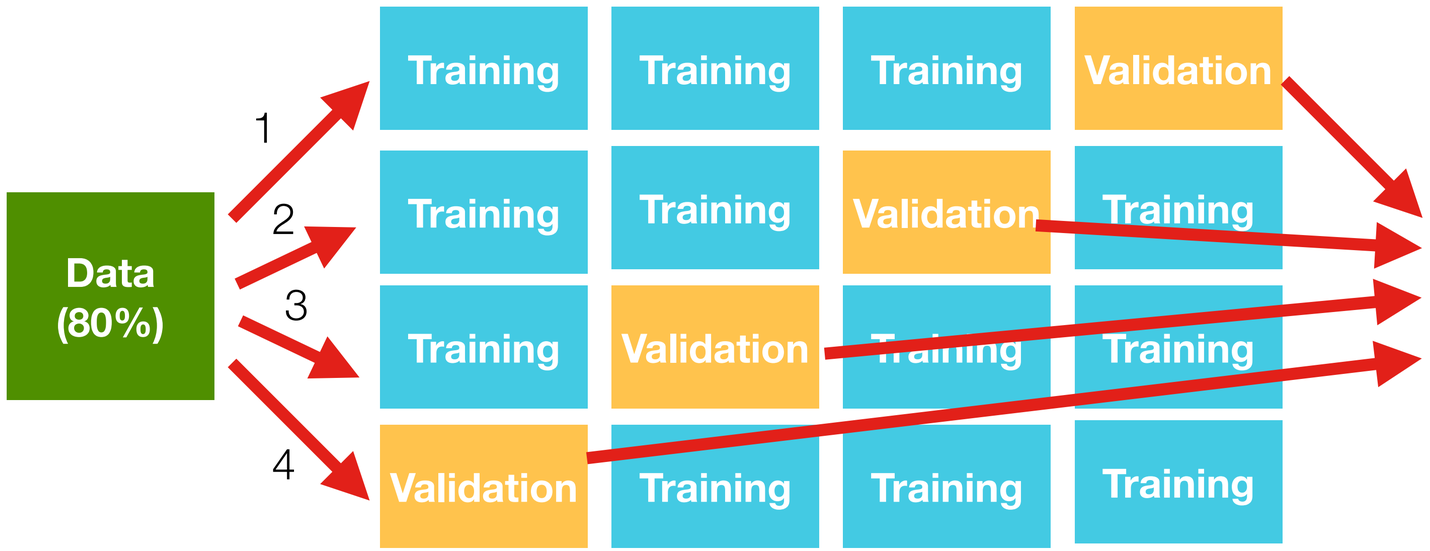



**NEVER use the test set for training —>  That is CHEATING!**

# Model Training: Cross-validation

- A way to use all the data for training and testing, by recycling it several times
  - Splitt the data in *n* portions
  - Train the model *n* times using *n-1* portions for training
- Useful when dataset is small



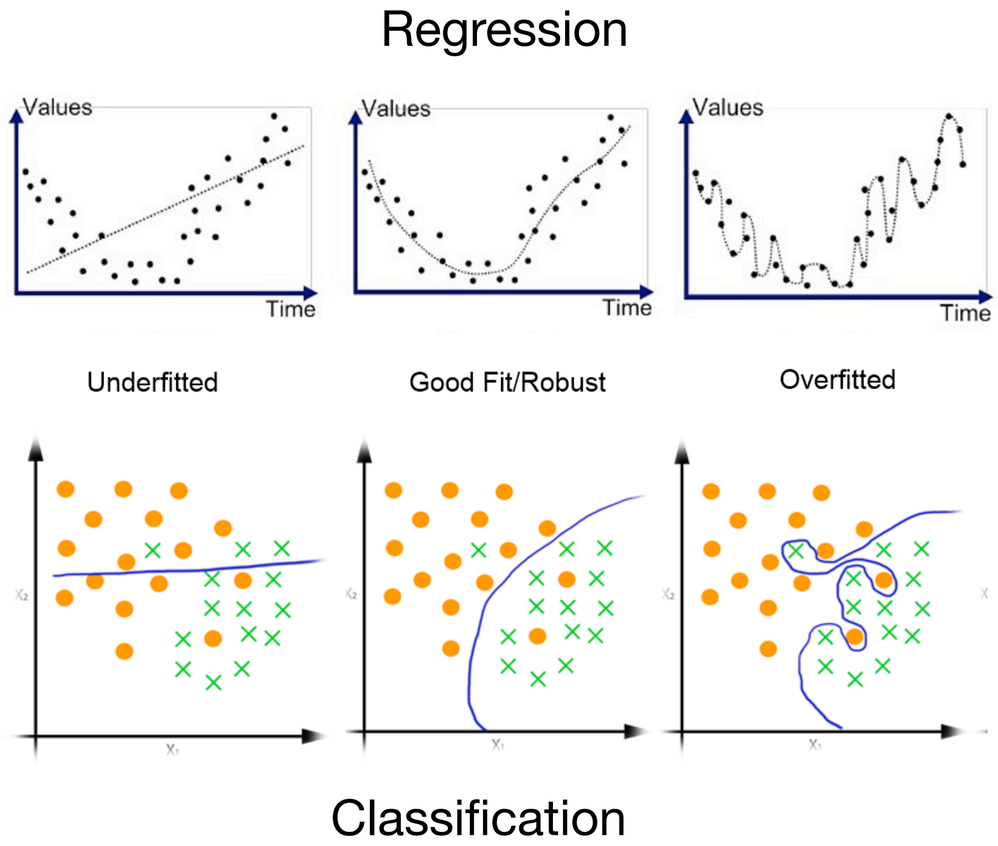

$$Score = \frac{Score_1 + Score_2 + Score_3 + Score + 4}{4}$$

**Example: four-fold cross-validation**

# No free-lunch

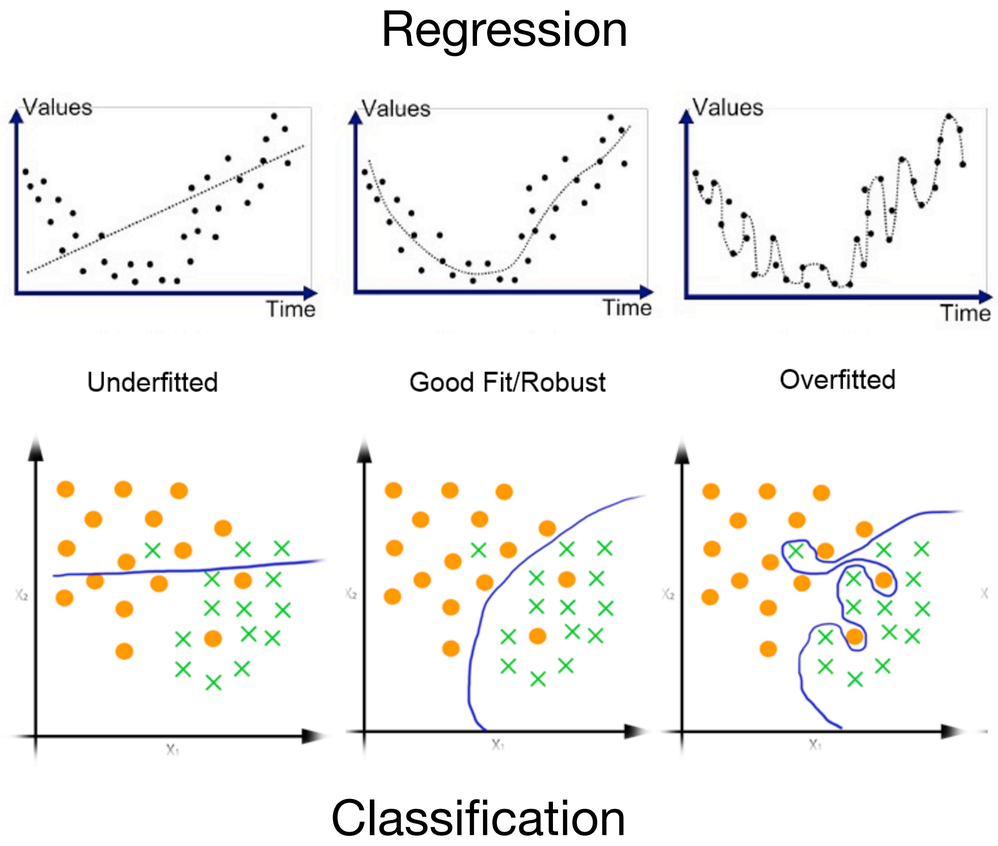
- There is no one best machine learning algorithm for all problems and datasets

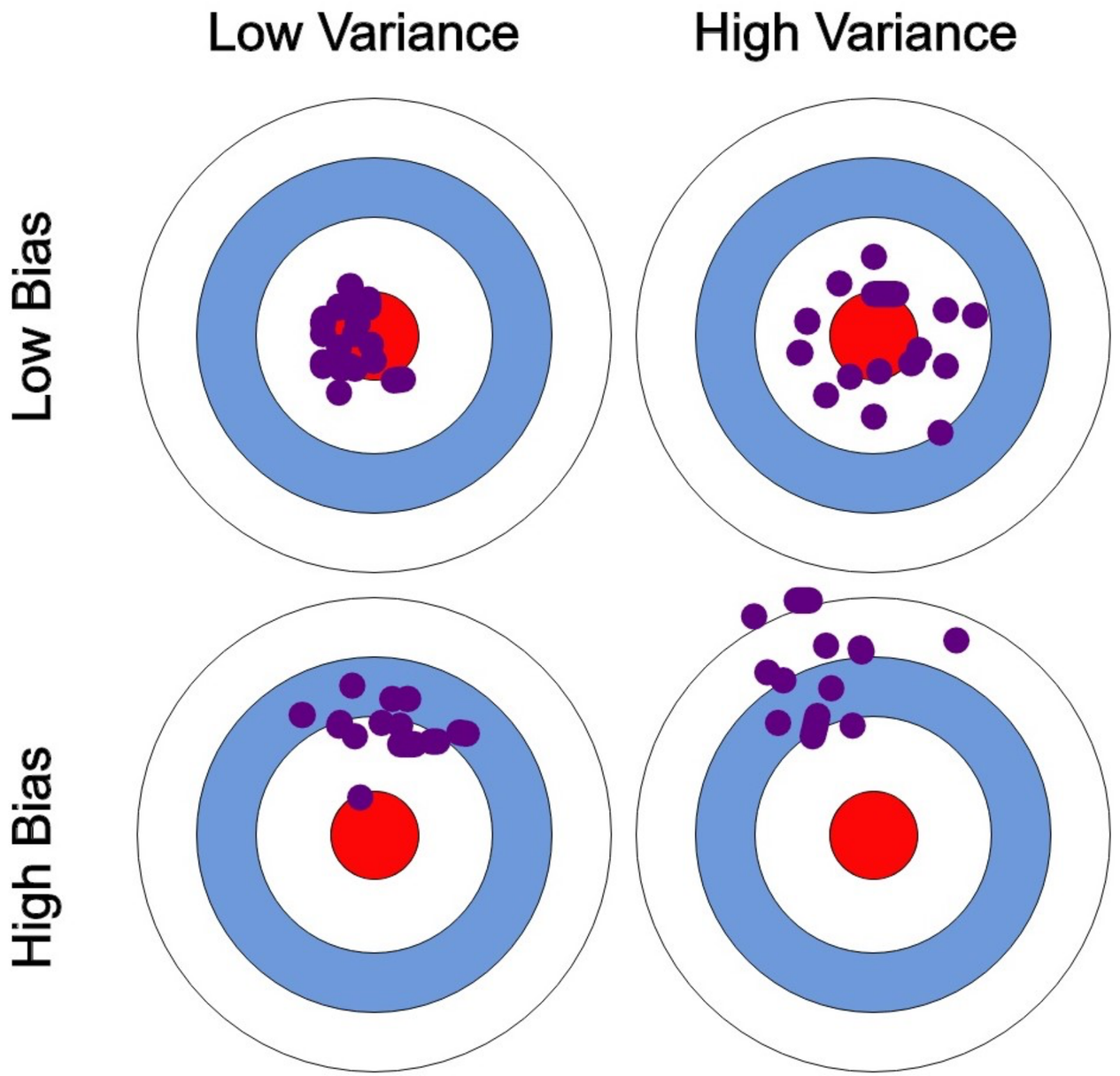
- **Generalisation**
  - How well does a learned model generalize from the data it was trained on to a new evaluation set?



# Generalisation

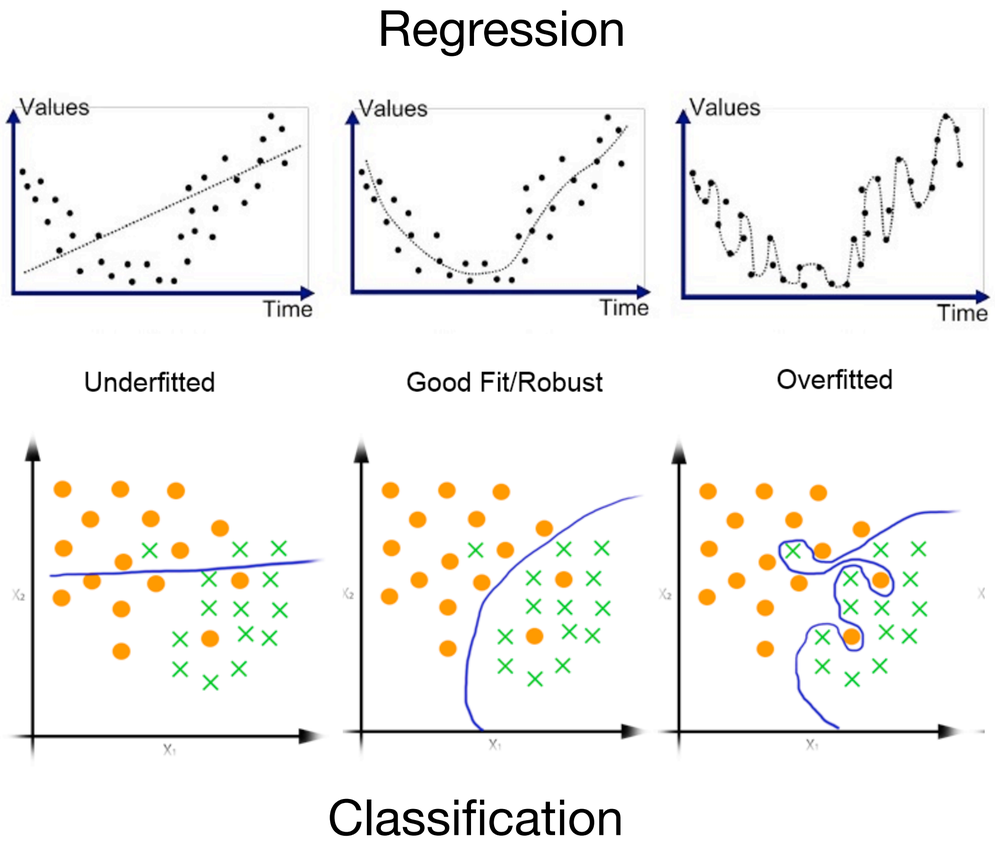
- **Challenge**: achieving good generalization and a small error rate
- Components of expected loss
  - **Noise** in our observations: **unavoidable**
  - **Bias**: how much the average model overall training sets differs from the true model
    - Error due to inaccurate assumptions/simplifications made by the model
  - **Variance**: how much models estimated from different training sets differ from each other
    - Too much sensitivity to the samples




Low Variance
High Variance
Low Bias
High Bias

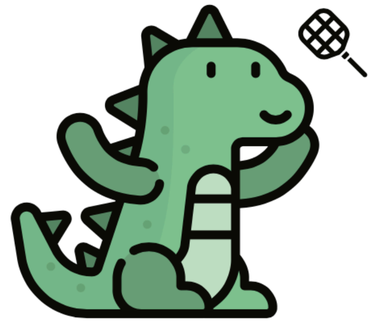
# Overfitting vs. Underfitting

- Protect against **overfitting**
  - learning a model that too closely matches the idiosyncrasies of the training data
  - model is too “complex” and fits irrelevant characteristics (noise) in the data
    - Low bias and high variance
    - Low training error and high test error
- Protect against **underfitting**
  - learning a model that does not adequately capture the patterns in the training data
  - The model is too “simple” to represent all the relevant class characteristics
    - High bias and low variance
    - High training error and high test error

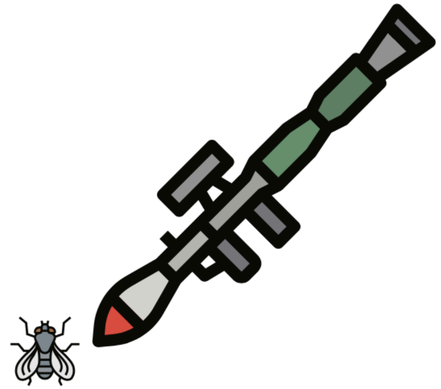


# Overfitting vs. Underfitting

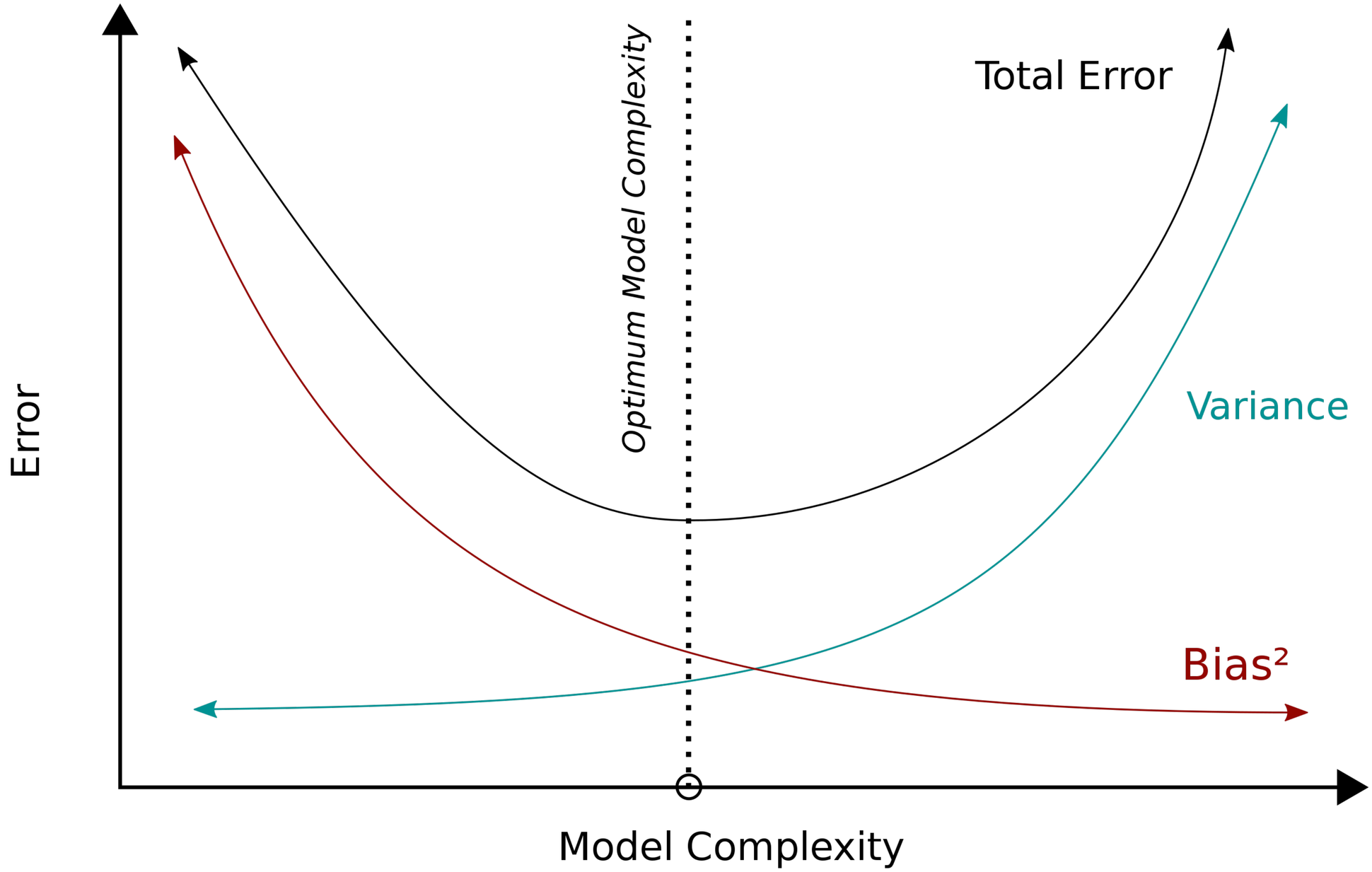
- Protect against **overfitting**
  - learning a model that too closely matches the idiosyncrasies of the training data
  - model is too “complex” and fits irrelevant characteristics (noise) in the data
    - Low bias and high variance
    - Low training error and high test error
- Protect against **underfitting**
  - learning a model that does not adequately capture the patterns in the training data
  - The model is too “simple” to represent all the relevant class characteristics
    - High bias and low variance
    - High training error and high test error



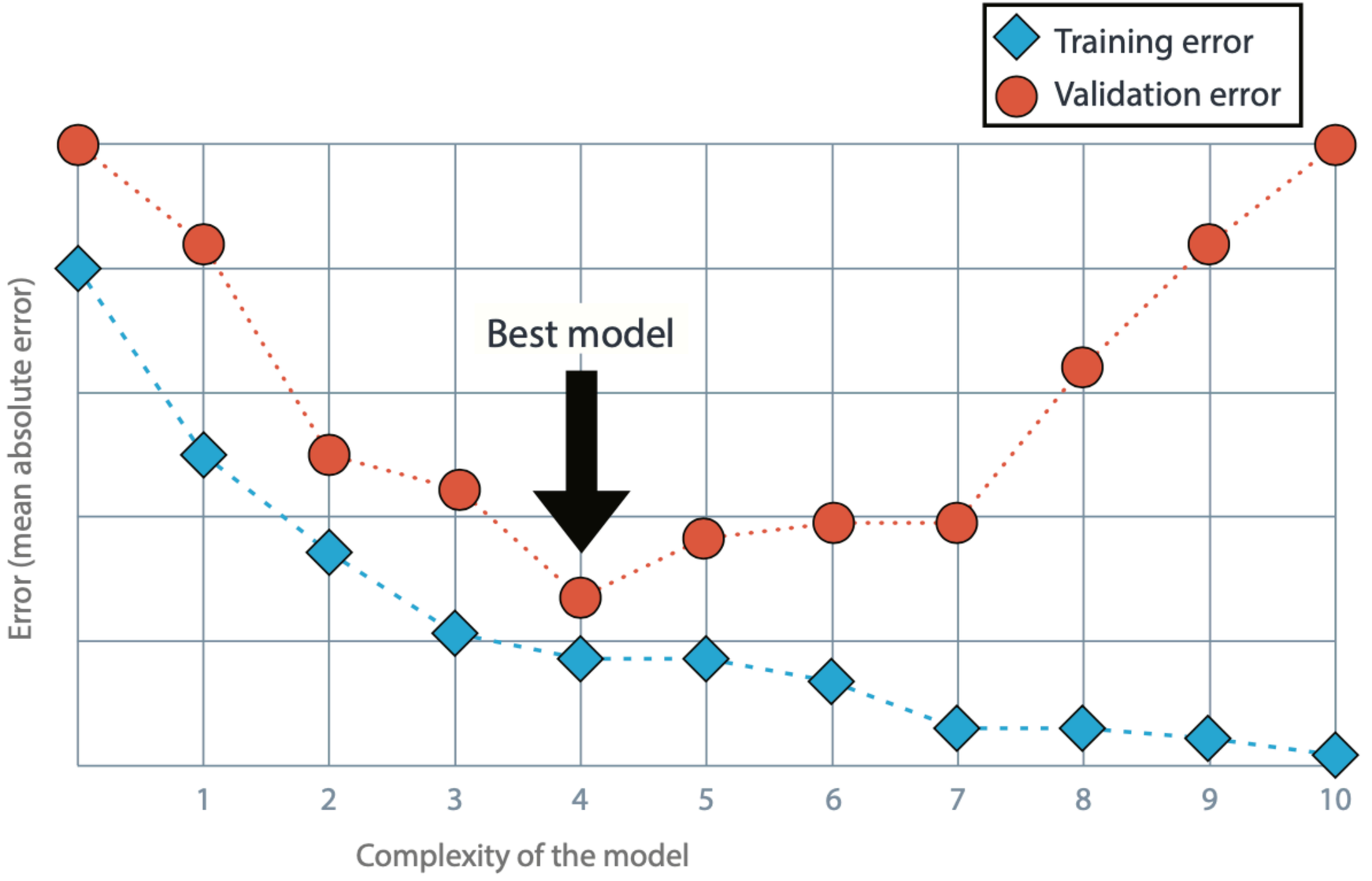
**Underfitting**



**Overfitting**

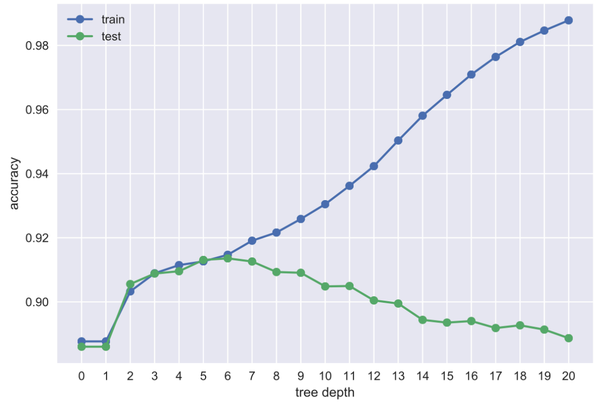



https://en.wikipedia.org/wiki/Bias–variance_tradeoff#/media/File:Bias_and_variance_contributing_to_total_error.svg


Training error
Validation error
Best model
Error (mean absolute error)
1
2
3
4
5
6
7
8
9
10
Complexity of the model

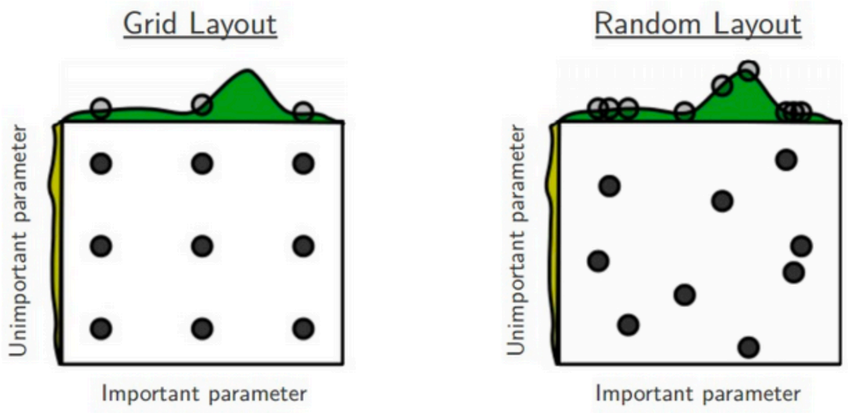
# Tuning Hyperparameters

- Inputs to the learning algorithms that control their behaviour
  - Examples:
    - maximum tree depth in decision trees
    - number of neighbours *k* in *k-nearest neighbour*
    - Neural networks: architecture, learning rate




- For a model to work well, they often need to be tuned carefully
  - Huge search space! may be inefficient to search exhaustively

- Possible approaches
  - **Grid search**: brute-force exhaustive search among a finite set of hyperparameter settings
    - All combinations are tried, then the best setting selected
  - **Random search**: for each hyperparameter, define a distribution (e.g. normal, uniform)
    - In the search loop, we sample randomly from these distributions

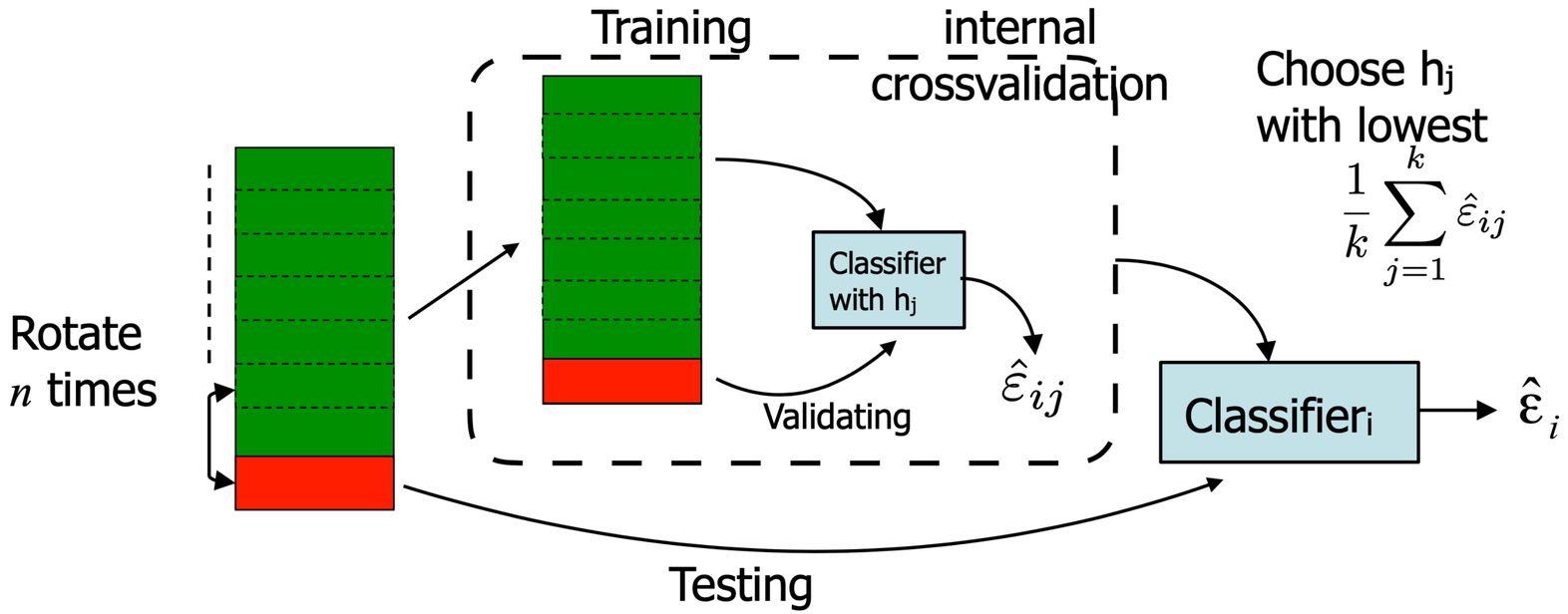



https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a

**DON'T optimise these numbers by looking at the test set!  That is CHEATING!**

# Double Cross-Validation

- To optimise over the hyperparameter do cross-validation inside another cross-validation
- The minimum error is often not the most interesting. Try to understand the advantages/disadvantages
  - What errors are made? (inspect objects, inspect labels)
  - What classes are problematic? (confusion matrix)
  - Does adding training data help? (learning curve)
  - How robust is the model?



# Model Evaluation: Experiments

- Compare to one or more baselines
  - Existing solution
  - Trivial (new) solution
  - Rule-based solution
  - Multiple ML models

**More next week**

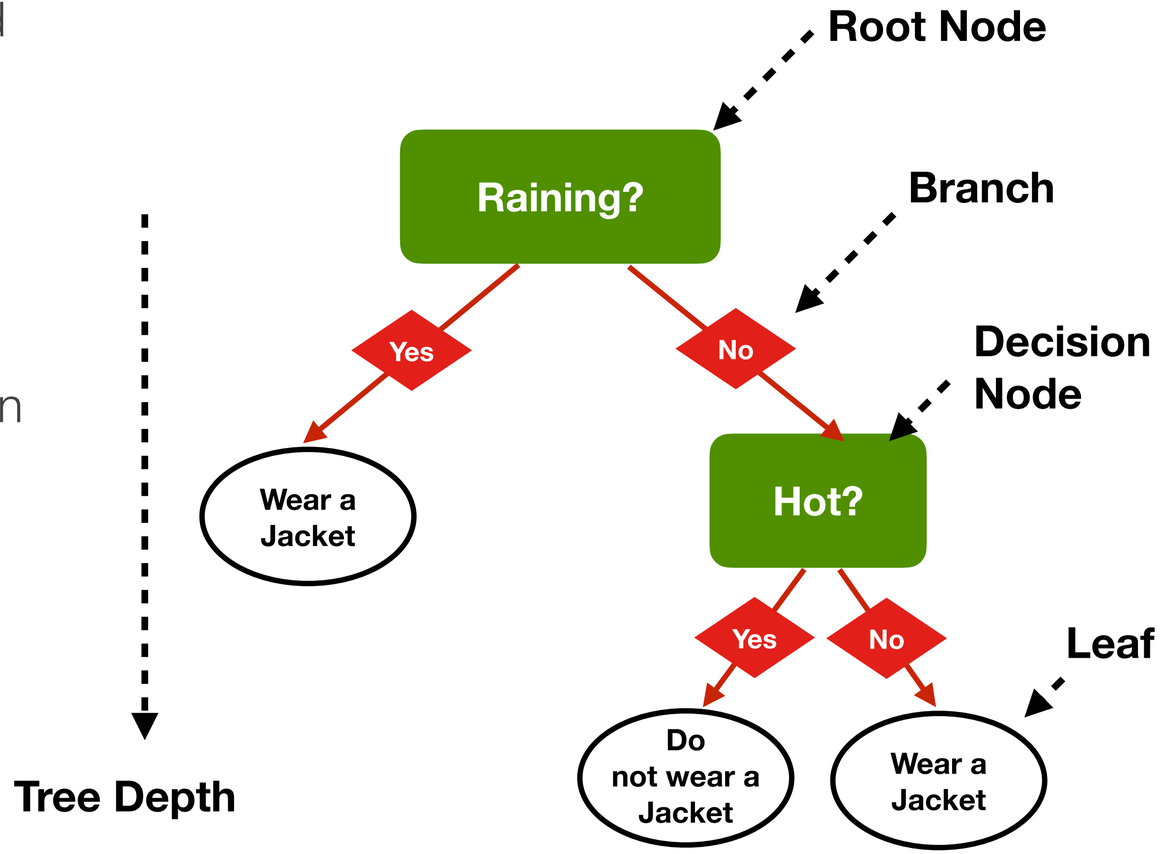
# Decision Trees

# Decision Trees

- Machine learning models used both for **classification** and **regression**
- Trained with labelled data (supervised learning)
  - classes —> classification
  - values —> regression

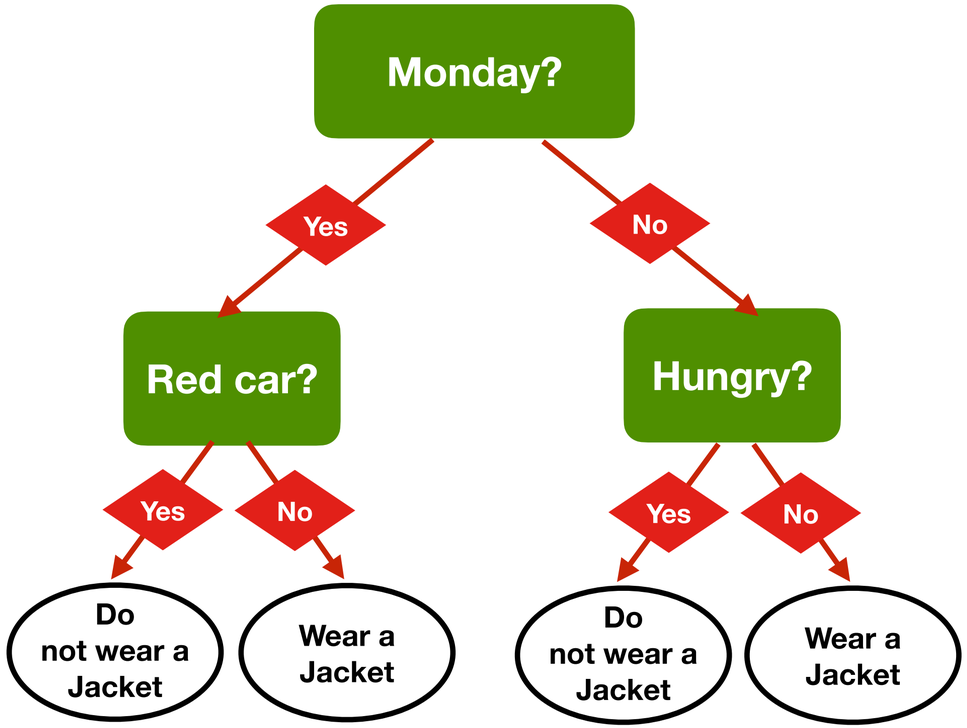
- A very simple model that resembles human reasoning when making predictions:
  - Answering a lot of yes/no questions based on feature values

- Problems:
  - Which questions to answer?
  - How many questions? (Tree depth)
  - In which order?



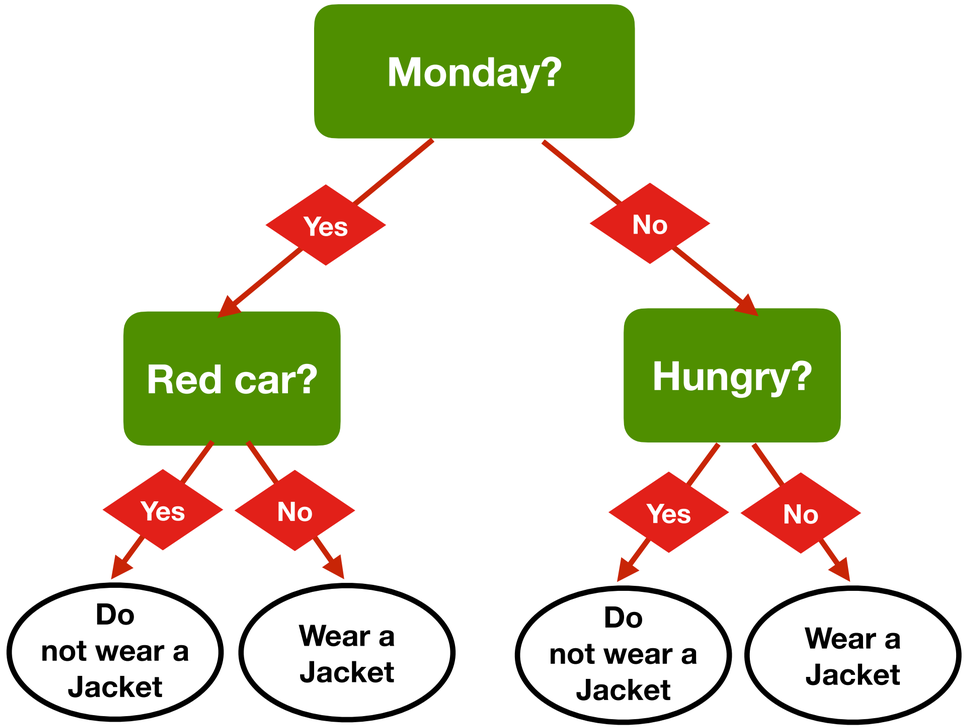
# Same Problem, Multiple Trees

- Feature space
  - Am I hungry?
  - Is there a red car outside?
  - Is it Monday?
  - Is it raining?
  - Is it cold outside?



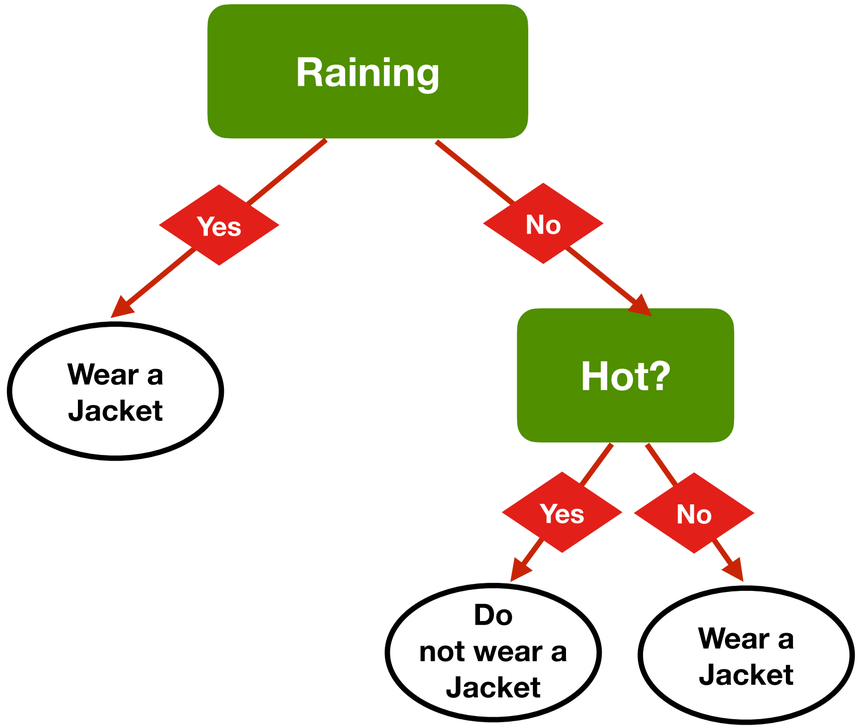
# Same Problem, Multiple Trees

- Feature space
  - ~~Am I hungry?~~
  - ~~Is there a red car outside?~~
  - ~~Is it Monday?~~
  - Is it raining?
  - Is it cold outside?

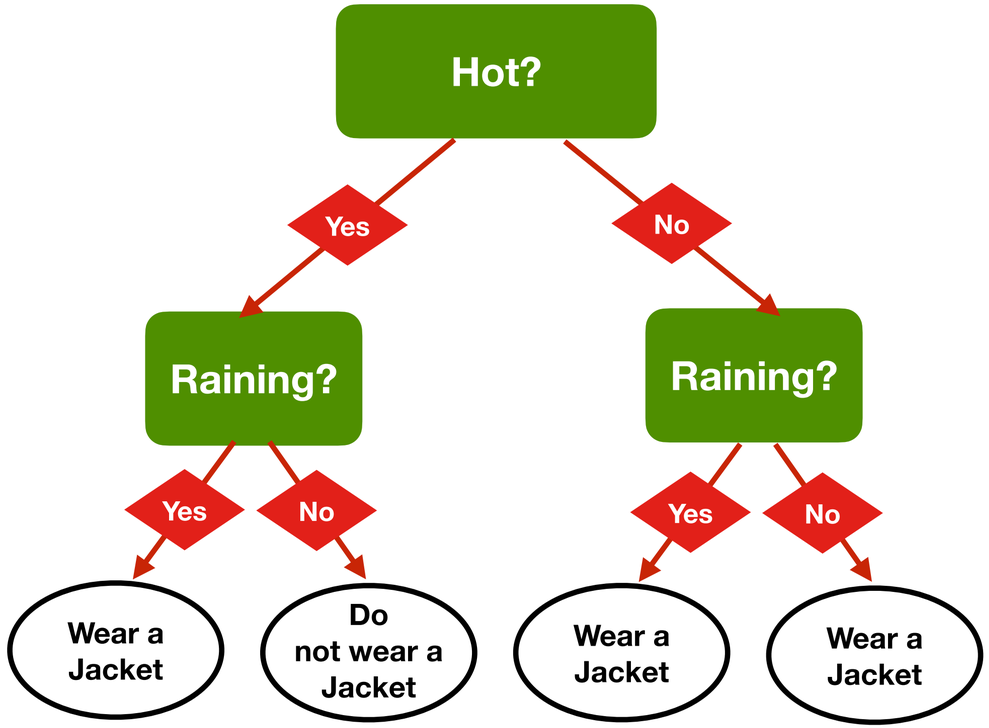


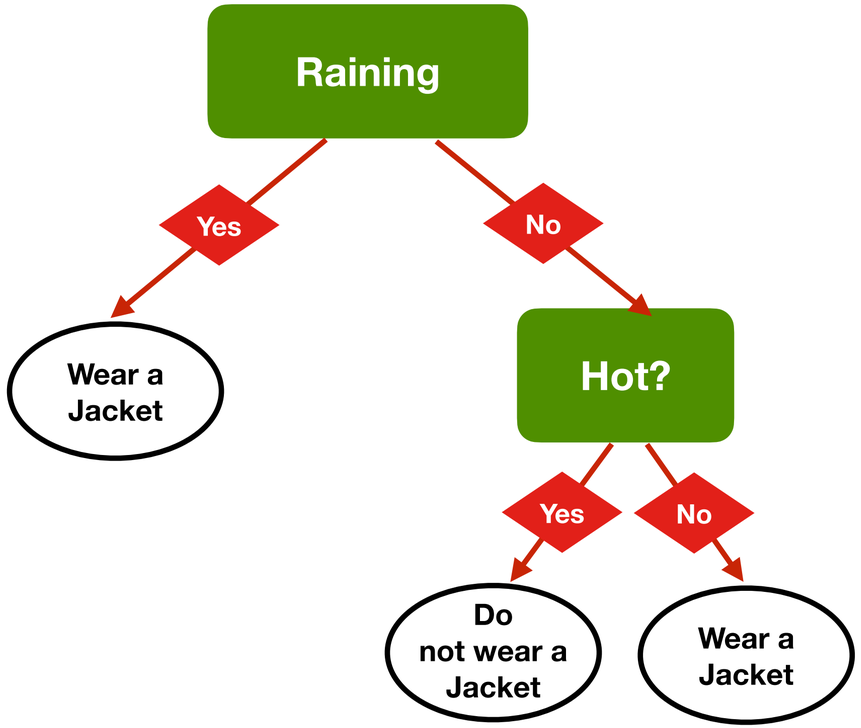
# Same Problem, Multiple Trees

- Feature space
  - ~~Am I hungry?~~
  - ~~Is there a red car outside?~~
  - ~~Is it Monday?~~
  - Is it raining?
  - Is it cold outside?

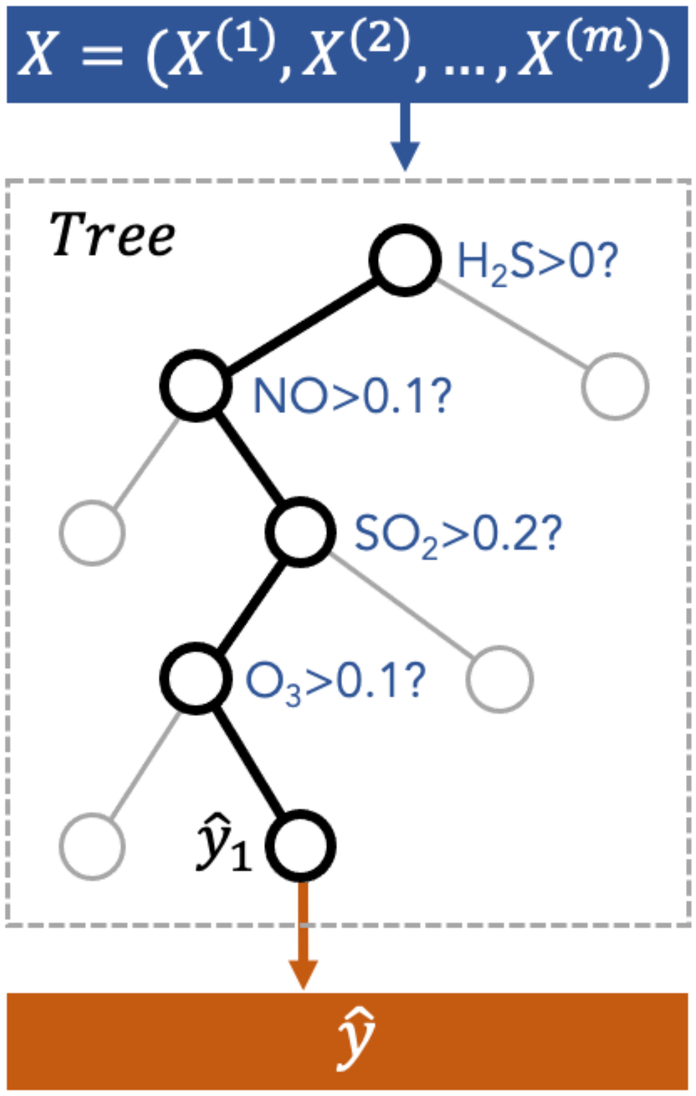


# Same decision, different trees



# Tutorial 3




PM, $SO_2$, CO, NO, $NO_2$, $O_3$, $H_2S$, and wind information

Prediction of bad smell (yes/no)

# How to decide the best question to ask?

- **3 metrics**
  - **Accuracy**
    - Which question helps me be **correct** more often?
  - **Gini Impurity Index**
    - A measure of *diversity* in a dataset —> diversity of classes in a given leaf node
      - *index = 0* means that all the items in a leaf node have the same class
    - Which question helps me obtain the lowest average **Gini impurity Index**?
  - **Entropy**
    - Another measure of *diversity* linked to information theory
    - Which question helps me obtain the lowest average **entropy**?

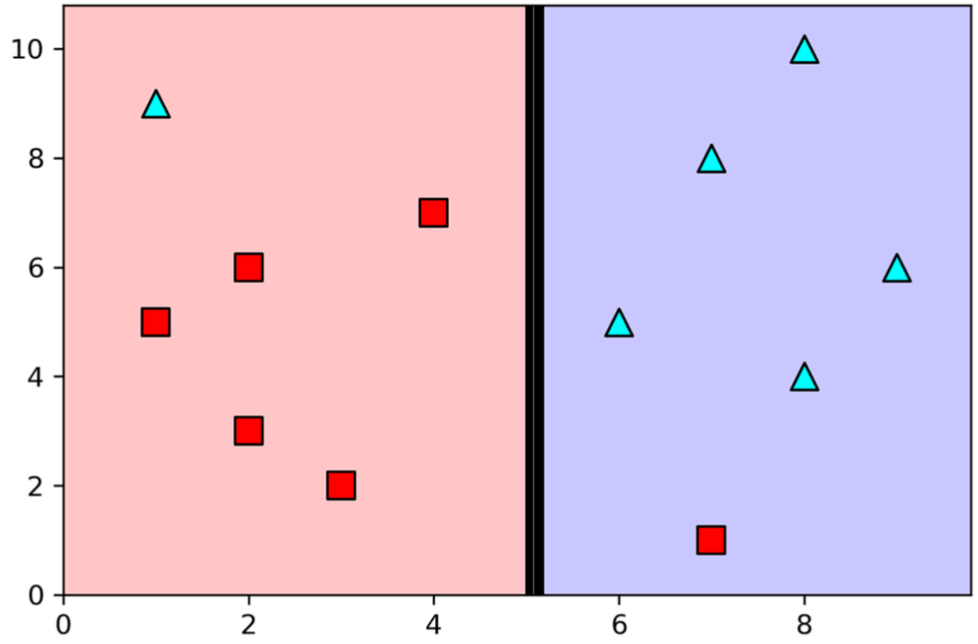
# Building the tree (pseudo-code)

- **Add a root node, and associate it with the entire dataset**
  - This node has level 0. Call it a leaf node

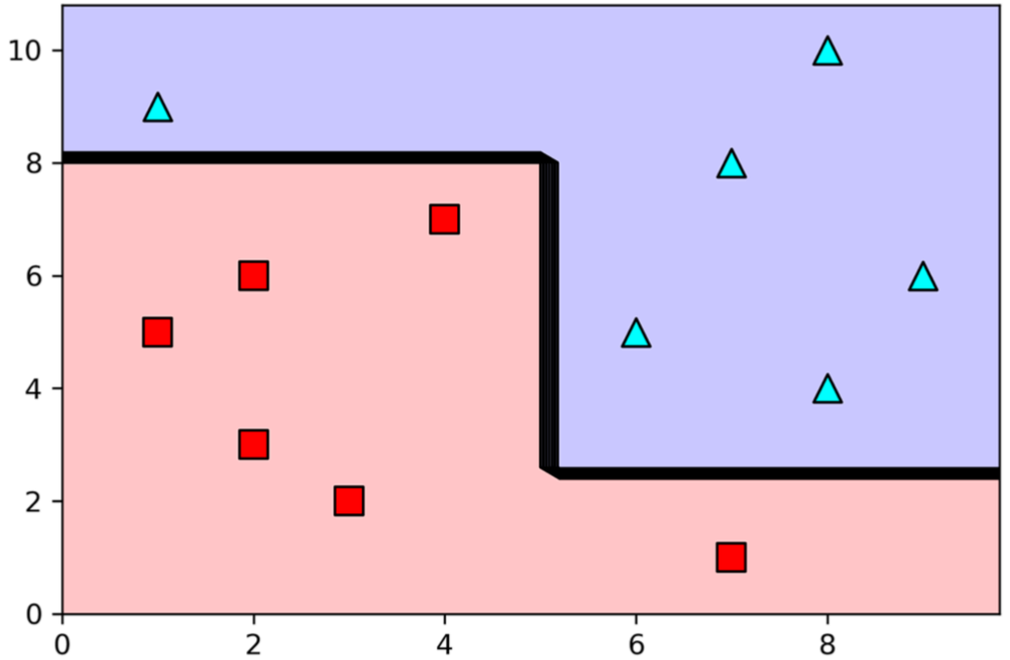
**Hyperparameter: tree depth**
***Stopping condition***

- **Repeat until the stopping conditions are met at every leaf node**
  - Pick one of the leaf nodes at the highest level
  - Go through all the features, and select the one that splits the samples corresponding to that node in an optimal way, according to the selected metric.
    - Associate that feature to the node
  - This feature splits the dataset into two branches
    - Create two new leaf nodes, one for each branch
    - Associate the corresponding samples to each of the nodes
  - If the stopping conditions allow a split, turn the node into a decision node, and add two new leaf nodes underneath it
    - If the level of the node is i, the two new leaf nodes are at level i + 1
  - If the stopping conditions don't allow a split, the node becomes a leaf node
    - Associate the most common label among its samples
    - That label is the prediction at the leaf

# A geometrical perspective




- Step 1 - Select the first question
- **X >= 5**
  - Best possible prediction accuracy with one feature




- Step 2 - Iterate
- **x < 5 & y< 8;  x >=5  & y>=2**
  - Perfect split of the feature space

# Decision Trees: Pros and Cons

- **PROs**
  - Simple to understand and to interpret. Trees can be visualised
  - Requires little data preparation. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed
  - Able to handle both numerical and categorical data

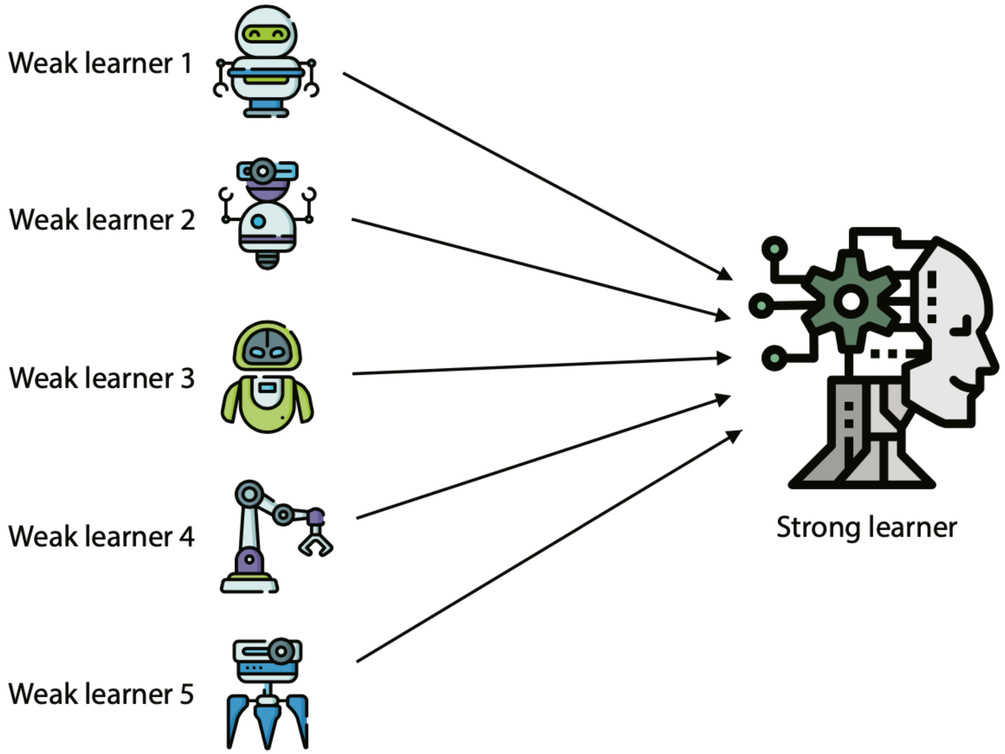
- **Cons**
  - Possible to create over-complex trees that do not generalise the data well
    - overfitting
  - Unstable —> small variations in the data might result in a completely different tree being generated
  - Biased trees if some classes dominate

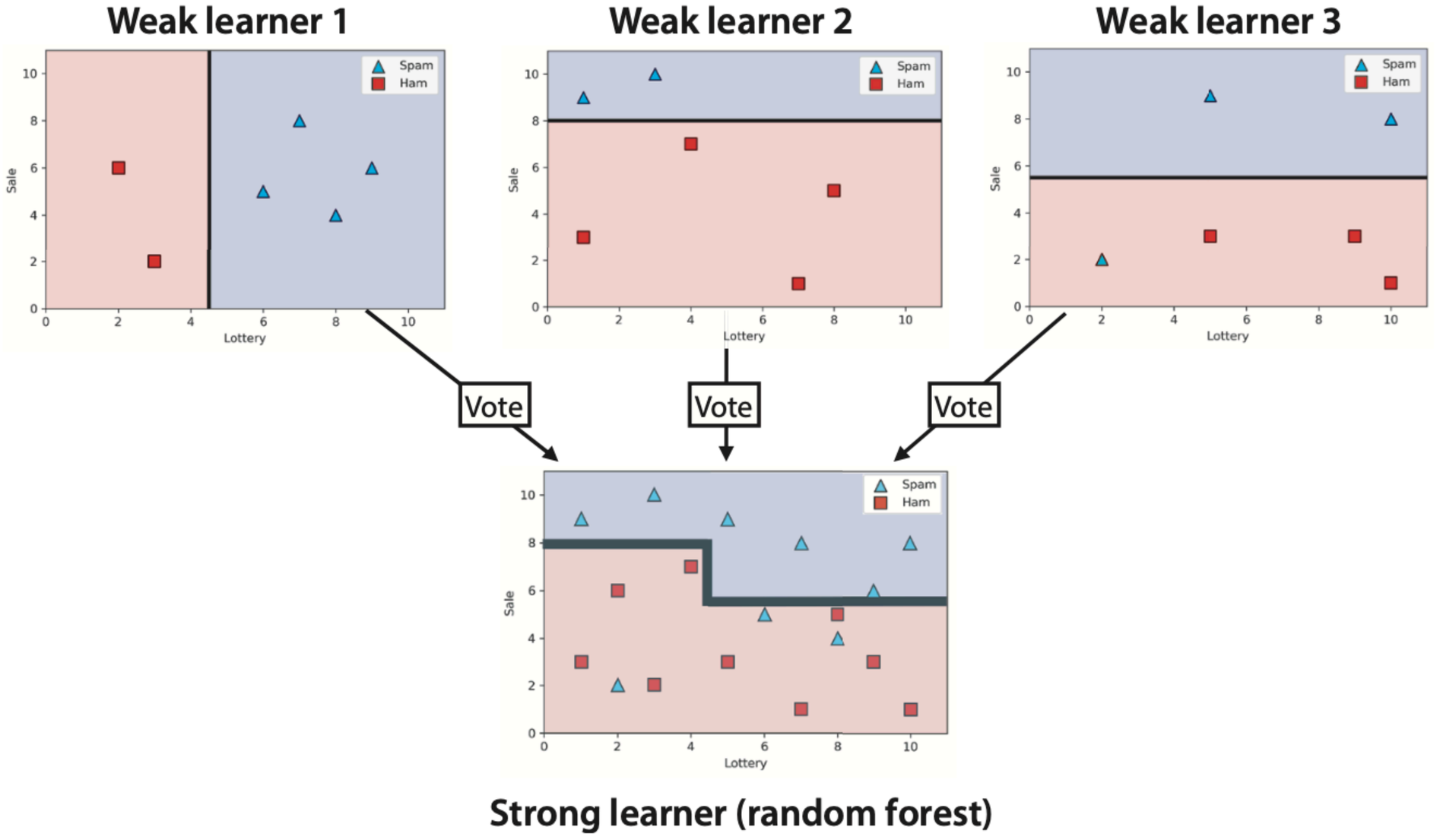
https://scikit-learn.org/stable/modules/tree.html

# Ensemble learning: Random Forest

- Idea: combine several “weak” learners to build a strong learner
  - Build random training sets from the dataset
  - Train a different model on each of the sets
    - weak learners
  - Combination the weak models by voting (if it is a classification model) or averaging the predictions (if it is a regression model)
    - For any input, each of the weak learners predicts a value
    - The most common output (or the average) is the output of the strong learner

- Random Forest
  - Weak learners are **decision trees**




Weak learner 1
Weak learner 2
Weak learner 3
Spam
Ham
Sale
Lottery
Vote
Vote
Vote
Strong learner (random forest)

# Machine Learning For Design

## Lecture 8 - Designing And Develop Machine Learning Models / Part 2

Alessandro Bozzon
23/03/2022

mlfd-io@tudelft.nl
www.ml4design.com

# Credits

- Grokking Machine Learning. Luis G. Serrano. Manning, 2021
- https://scikit-learn.org/stable/modules/tree.html
- CIS 419/519 Applied Machine Learning. Eric Eaton, Dinesh Jayaraman. https://www.seas.upenn.edu/~cis519/spring2020/