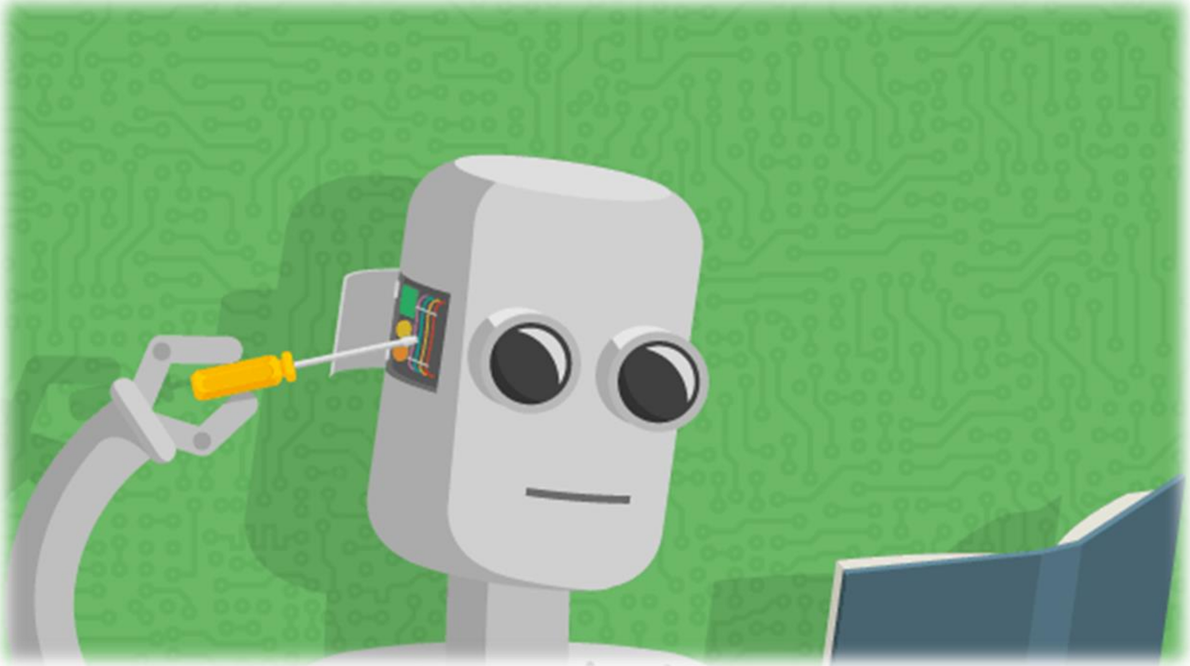
Refresher on Machine Learning

Alberto TONDA, Ph.D. (Senior permanent researcher, DR)

UMR 518 MIA-PS, INRAE, AgroParisTech, Université Paris-Saclay
UAR 3611, Institut des Systèmes Complexes de Paris Île-de-France

Outline

- Artificial Intelligence
- Machine Learning
- Typical ML pipeline
- Evaluating performance
- Overfitting



What is Artificial Intelligence?

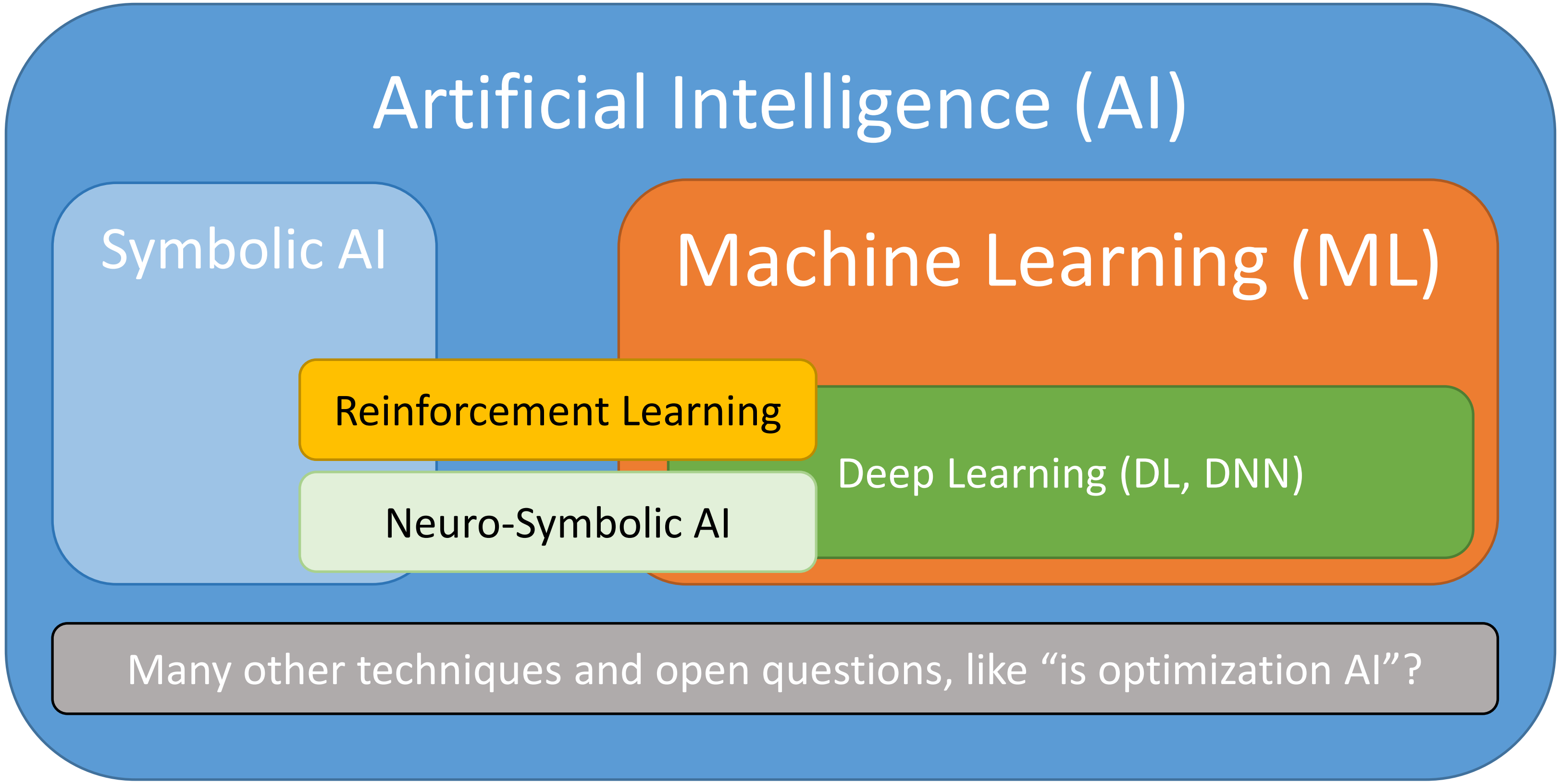
What is Artificial Intelligence?

- John McCarthy, one of the founding fathers of Artificial Intelligence
- *«I invented this term Artificial Intelligence [...] because [...] we were* ***trying to get money*** *for a summer study [...]»*

What is Artificial Intelligence?

- Short answer, there is no clear definition
  - We do not have a good definition of *intelligence*, so...
  - Broadly speaking, AI defines a *field* more than a *method*
- Tentative definitions (there is no agreement)
  - «When a non-biological being successfully completes a task commonly believed to require biological intelligence»
  - «Perceiving, synthesizing, and inferring information»
  - «Efficiency and speed, in learning a new task» (Chollet, 2019)
- How do we *measure* intelligence?

What is Artificial Intelligence?


Artificial Intelligence (AI)
Symbolic AI
Machine Learning (ML)
Reinforcement Learning
Deep Learning (DL, DNN)
Neuro-Symbolic AI
Many other techniques and open questions, like “is optimization AI”?

Most successful approaches employ **MULTIPLE TECHNIQUES AT THE SAME TIME**

What is Artificial Intelligence?

NARROW / WEAK

Focused on a specific task

- Symbolic AI
  - E.g. rule-based systems
- Machine learning
  - Supervised, unsupervised
  - Natural language processing
  - Image recognition/segmentation
- Reinforcement learning
- Neuro-symbolic AI

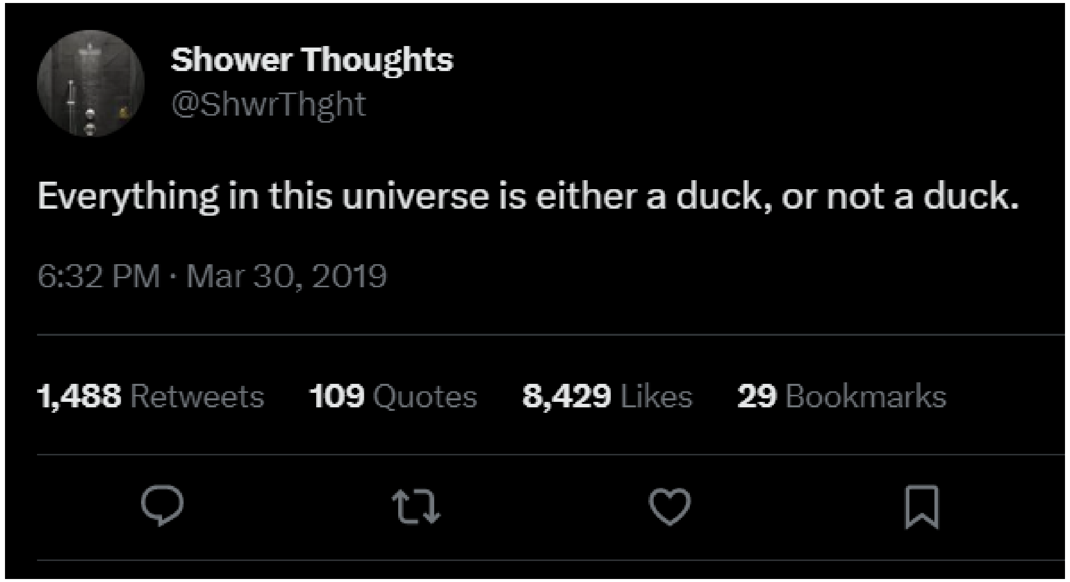
GENERAL (AGI)

Can perform any type of (human?) task

- Does not exist (...yet?)
- Closest thing is NLP: Large Language Models (LLM) like ChatGPT

Symbolic AI

- Symbolic manipulation
  - Reality is *continuous* (with good approximation)
  - Symbols are *discrete*, and humans are good at using them



Shower Thoughts
@ShwrThght

Everything in this universe is either a duck, or not a duck.

6:32 PM · Mar 30, 2019

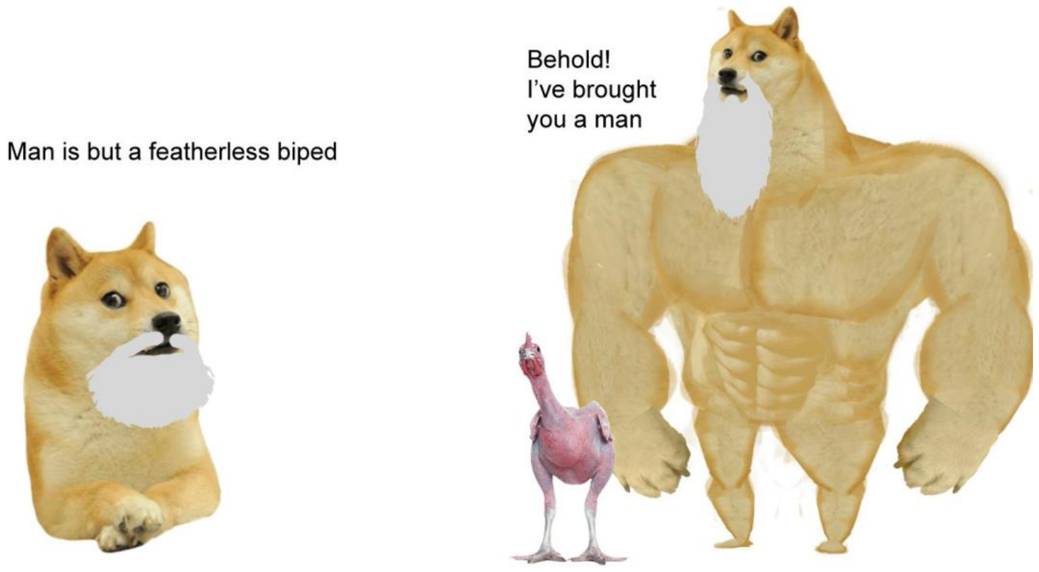
1,488 Retweets 109 Quotes 8,429 Likes 29 Bookmarks

Symbolic AI

- Symbols seem normal and natural, map into the real world (in linguistics, it’s called *extension*)
- Natural language is a powerful human symbol manipulator
- However, there is chaos hidden under the surface
    - What is the reality of a *river*?
    - What is the reality of a *chair*?
    - What is the reality of a *number*?

Symbolic AI

- Symbol can be hard to define, but we grasp it intuitively
  - It’s an old, **old** problem: see Plato and Diogenes
  - *Entire fields of research* on this (neuroscience, cognitive sciences, neurolinguistics, …)
- “Explaining” symbols to AI is harder yet
- Issues with “common sense”
- Reached limits in the 1980s



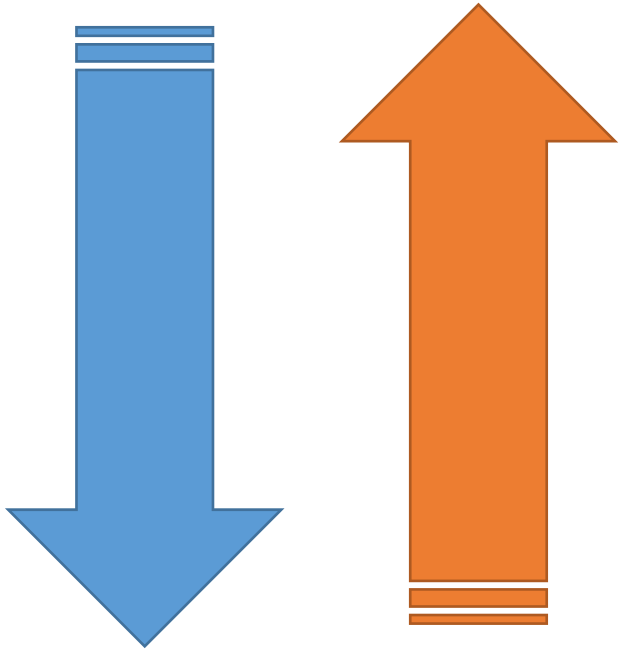
Symbolic AI

- In practice, find or exploit human-readable rules
  - Expert systems (“if-then-else” rules)
  - Knowledge graphs, linking entities with relationships; ontologies
  - First-order logic rules
  - **Decision trees** (that are also considered part of ML, as they can be built automatically from data)

- Before the advent of ML, considerable success stories
- Symbolic AI is still in use, paired with ML

Rule-based AI vs Machine Learning?

Rules



Deduction

Induction

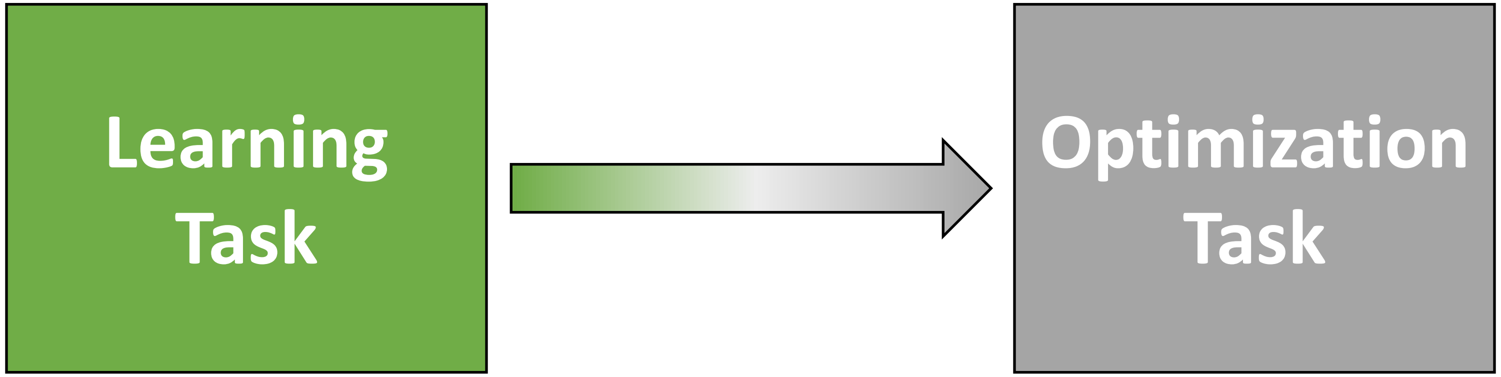
Samples

Machine Learning

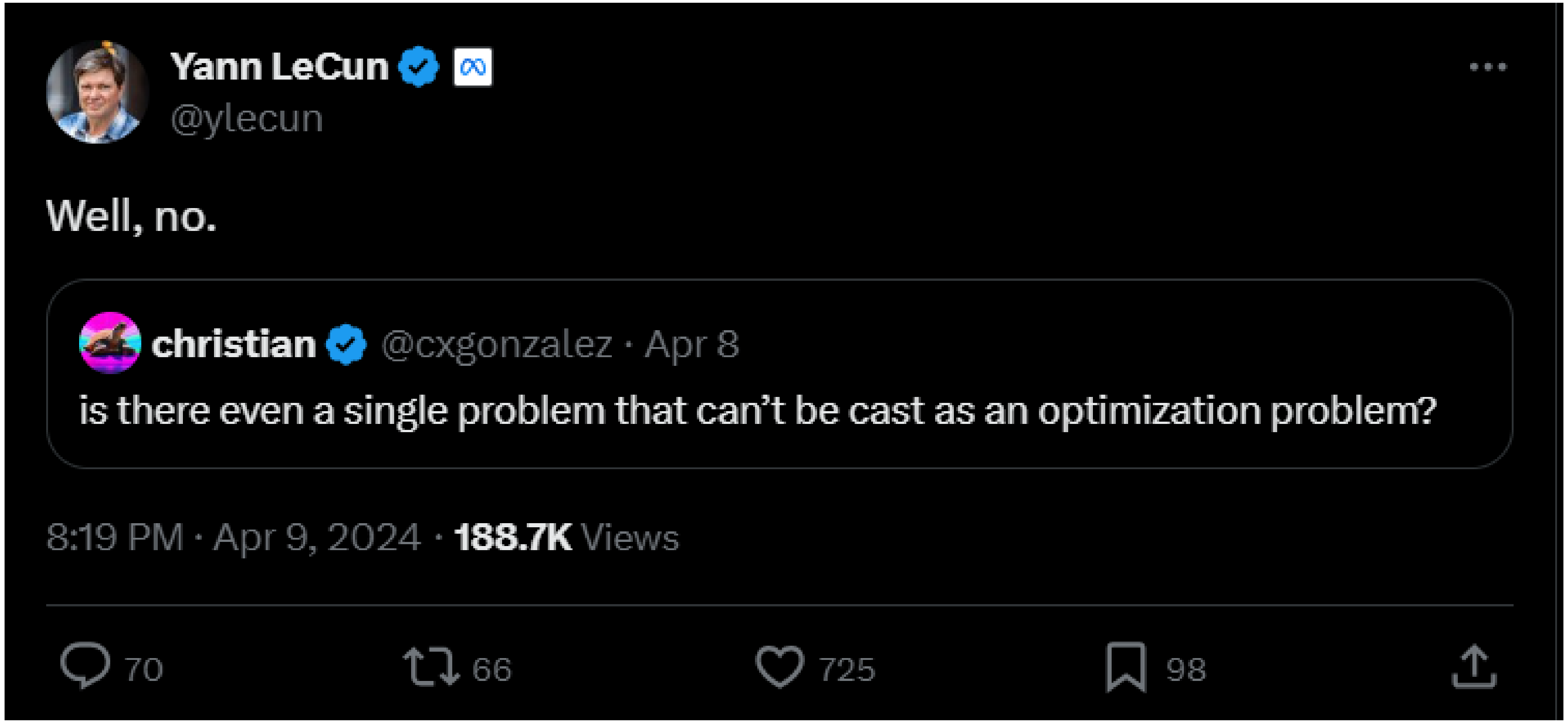
*Given a class of tasks **T**,*
*a performance measure **P**, and experience **E**,*
*a machine learning algorithm improves its performance measured with **P**, for tasks in **T**,*
*using the experience **E***

Machine Learning

- Learn a task directly from examples (induction)
  - No need for theory, just large quantities of data
  - *Samples* (rows) and *features* (columns)
- “Dirty secret” of ML: it’s mostly **optimization**
  - Restate **learning task** as **optimization task**
  - Solve it relying on available (training) data



So, machine learning is mostly optimization?



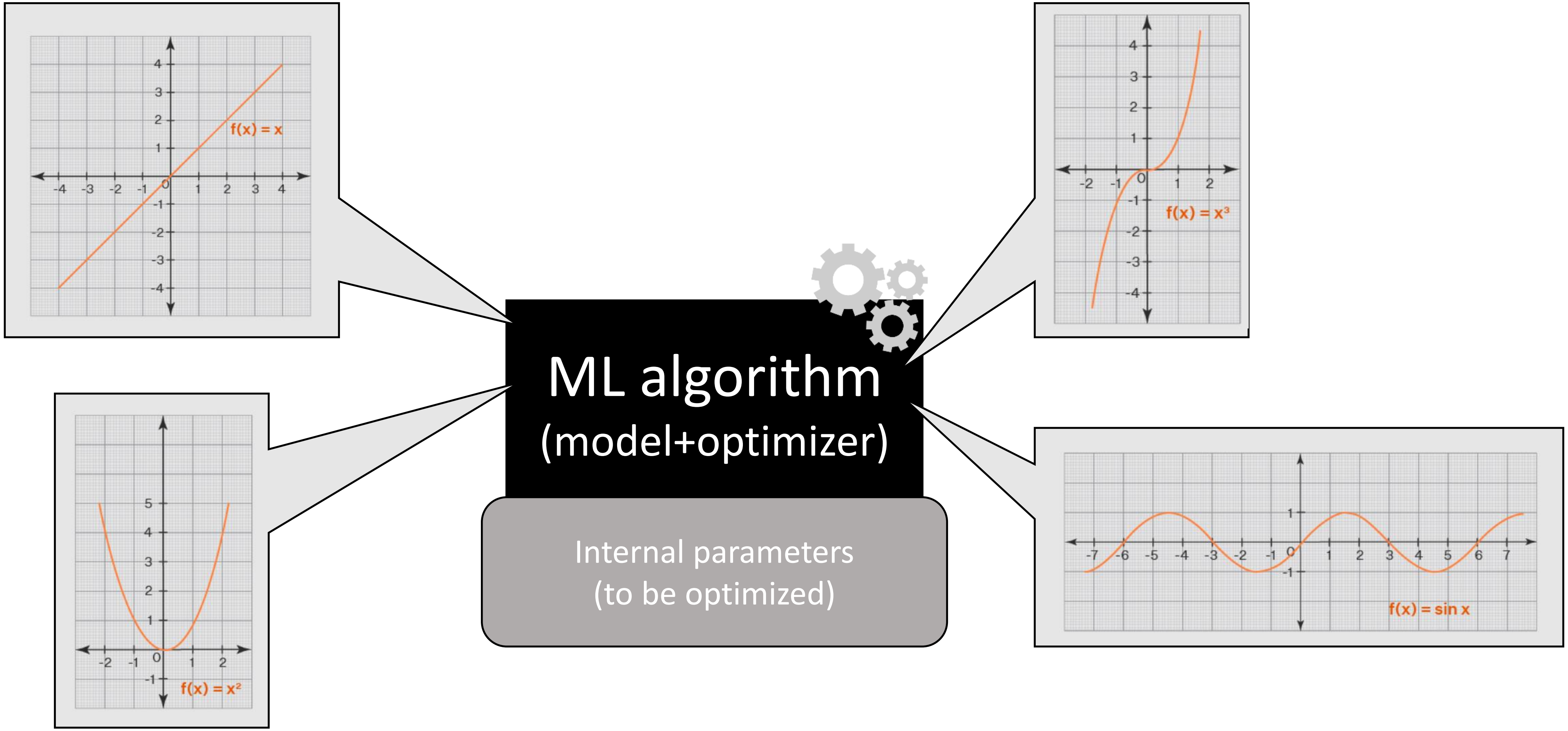
Yann LeCun
@ylecun

Well, no.

christian @cxgonzalez · Apr 8
is there even a single problem that can’t be cast as an optimization problem?

8:19 PM · Apr 9, 2024 · **188.7K** Views

Machine Learning algorithms



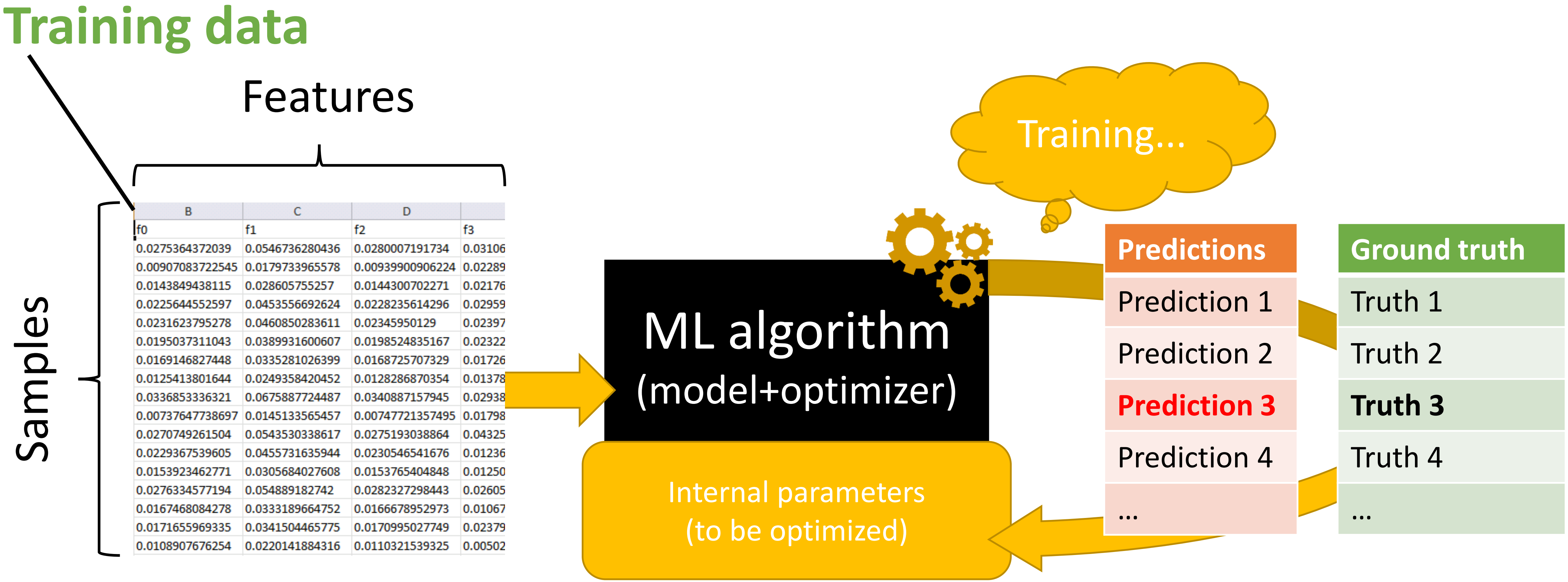
ML algorithm
(model+optimizer)

Internal parameters
(to be optimized)

Supervised Machine Learning

- Learn from examples for which correct answer is known
  - Data contains measured values of the target (**ground truth**)
  - Minimize difference between model predictions and ground truth
- Regression
  - Target is a continuous value (0.9, 22.5, 0.0017, …)
  - From the values of the features of a sample, **predict target value**
- Classification
  - Target is a category (good/bad, high/medium/low, toxic/ok, …)
  - From the values of the features of a sample, **assign to category**

Machine Learning (supervised)



B	C	D	
f0	f1	f2	f3
0.0275364372039	0.0546736280436	0.0280007191734	0.03106
0.00907083722545	0.0179733965578	0.00939900906224	0.02289
0.0143849438115	0.028605755257	0.0144300702271	0.02176
0.0225644552597	0.0453556692624	0.0228235614296	0.02959
0.0231623795278	0.0460850283611	0.02345950129	0.02397
0.0195037311043	0.0389931600607	0.0198524835167	0.02322
0.0169146827448	0.0335281026399	0.0168725707329	0.01726
0.0125413801644	0.0249358420452	0.0128286870354	0.01378
0.0336853336321	0.0675887724487	0.0340887157945	0.02938
0.00737647738697	0.0145133565457	0.00747721357495	0.01798
0.0270749261504	0.0543530338617	0.0275193038864	0.04325
0.0229367539605	0.0455731635944	0.0230546541676	0.01236
0.0153923462771	0.0305684027608	0.0153765404848	0.01250
0.0276334577194	0.054889182742	0.0282327298443	0.02605
0.0167468084278	0.0333189664752	0.0166678952973	0.01067
0.0171655969335	0.0341504465775	0.0170995027749	0.02379
0.0108907676254	0.0220141884316	0.0110321539325	0.00502

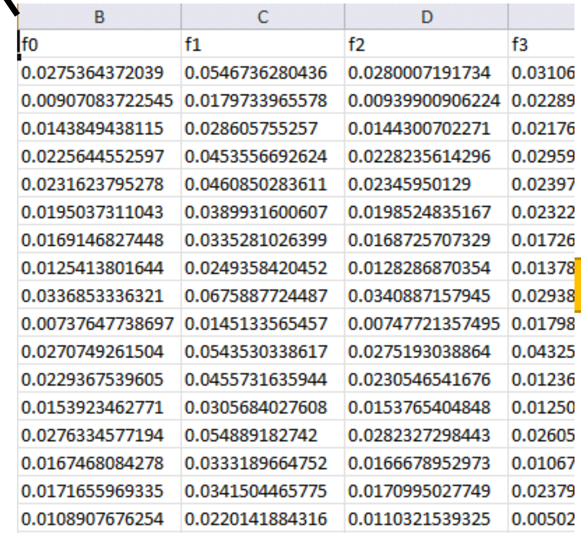
Predictions	Ground truth
Prediction 1	Truth 1
Prediction 2	Truth 2
Prediction 3	**Truth 3**
Prediction 4	Truth 4
…	…

Machine Learning (supervised)

Training data

Features

Samples

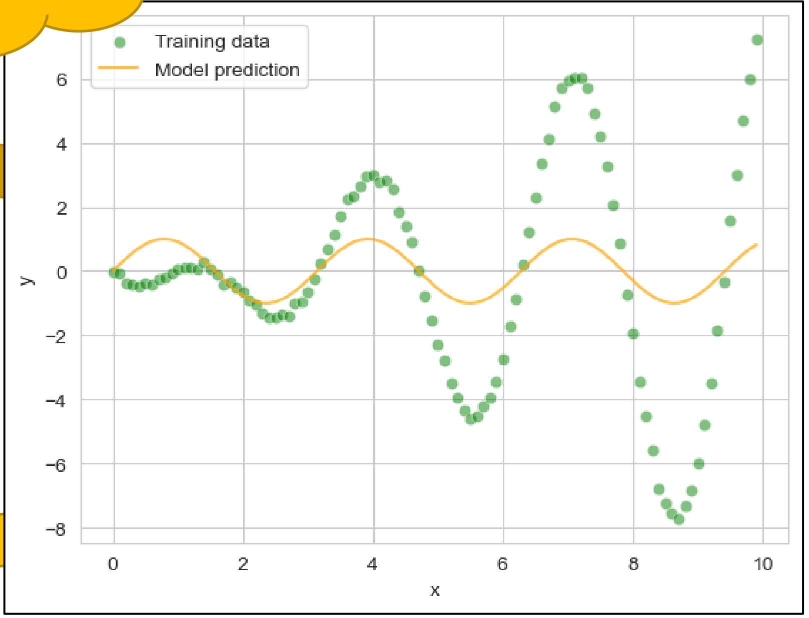


B	C	D	
f0	f1	f2	f3
0.0275364372039	0.0546736280436	0.0280007191734	0.03106
0.00907083722545	0.0179733965578	0.00939900906224	0.02289
0.0143849438115	0.028605755257	0.0144300702271	0.02176
0.0225644552597	0.0453556692624	0.0228235614296	0.02959
0.0231623795278	0.0460850283611	0.02345950129	0.02397
0.0195037311043	0.0389931600607	0.0198524835167	0.02322
0.0169146827448	0.0335281026399	0.0168725707329	0.01726
0.0125413801644	0.0249358420452	0.0128286870354	0.01378
0.0336853336321	0.0675887724487	0.0340887157945	0.02938
0.00737647738697	0.0145133565457	0.00747721357495	0.01798
0.0270749261504	0.0543530338617	0.0275193038864	0.04325
0.0229367539605	0.0455731635944	0.0230546541676	0.01236
0.0153923462771	0.0305684027608	0.0153765404848	0.01250
0.0276334577194	0.054889182742	0.0282327298443	0.02605
0.0167468084278	0.0333189664752	0.0166678952973	0.01067
0.0171655969335	0.0341504465775	0.0170995027749	0.02379
0.0108907676254	0.0220141884316	0.0110321539325	0.00502

Training...

ML algorithm
(model+optimizer)

Internal parameters
(to be optimized)

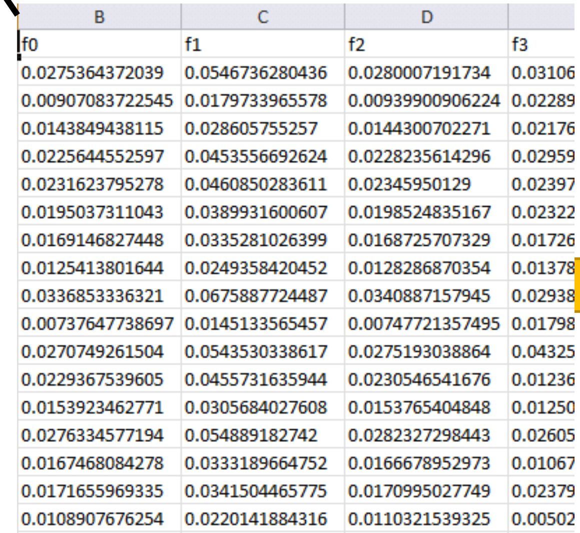


Machine Learning (supervised)

Training data

Features

Samples

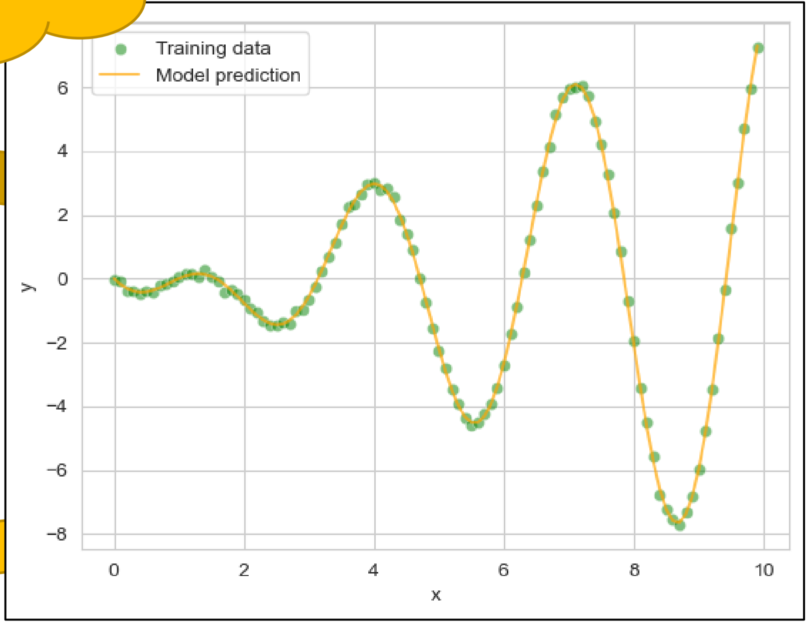


B	C	D	
f0	f1	f2	f3
0.0275364372039	0.0546736280436	0.0280007191734	0.03106
0.00907083722545	0.0179733965578	0.00939900906224	0.02289
0.0143849438115	0.028605755257	0.0144300702271	0.02176
0.0225644552597	0.0453556692624	0.0228235614296	0.02959
0.0231623795278	0.0460850283611	0.02345950129	0.02397
0.0195037311043	0.0389931600607	0.0198524835167	0.02322
0.0169146827448	0.0335281026399	0.0168725707329	0.01726
0.0125413801644	0.0249358420452	0.0128286870354	0.01378
0.0336853336321	0.0675887724487	0.0340887157945	0.02938
0.00737647738697	0.0145133565457	0.00747721357495	0.01798
0.0270749261504	0.0543530338617	0.0275193038864	0.04325
0.0229367539605	0.0455731635944	0.0230546541676	0.01236
0.0153923462771	0.0305684027608	0.0153765404848	0.01250
0.0276334577194	0.054889182742	0.0282327298443	0.02605
0.0167468084278	0.0333189664752	0.0166678952973	0.01067
0.0171655969335	0.0341504465775	0.0170995027749	0.02379
0.0108907676254	0.0220141884316	0.0110321539325	0.00502

ML algorithm
(model+optimizer)

Training...

Internal parameters
(to be optimized)

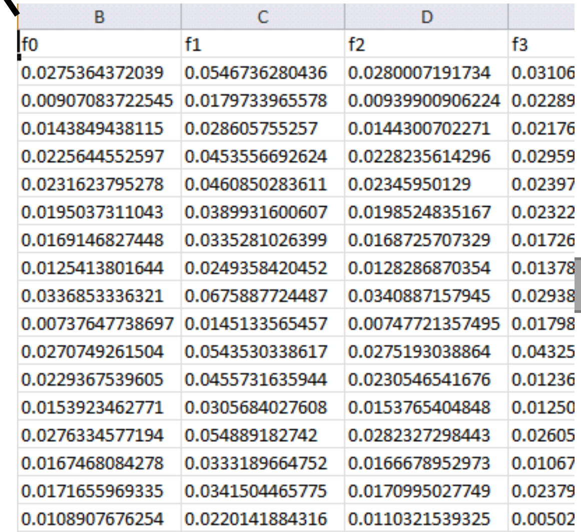


Machine Learning (supervised)

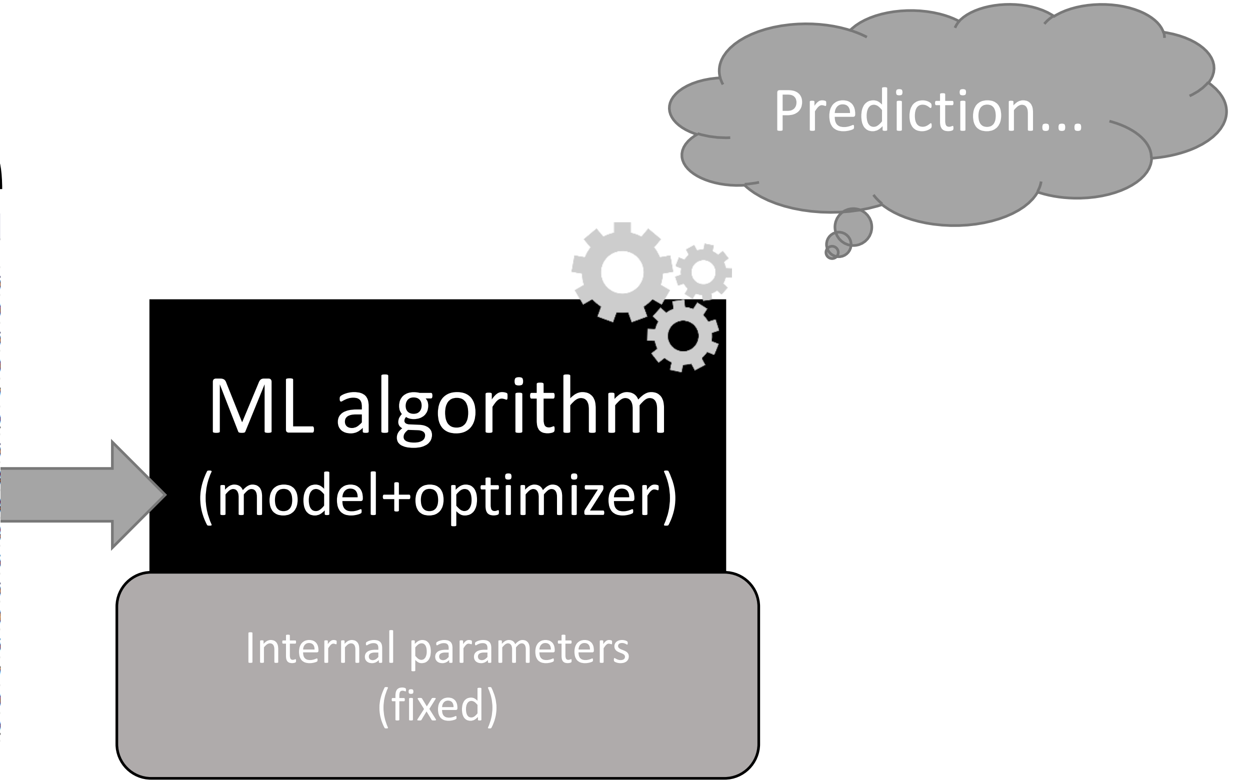
Test (unseen) data

Features

Samples



B	C	D	
f0	f1	f2	f3
0.0275364372039	0.0546736280436	0.0280007191734	0.03106
0.00907083722545	0.0179733965578	0.00939900906224	0.02289
0.0143849438115	0.028605755257	0.0144300702271	0.02176
0.0225644552597	0.0453556692624	0.0228235614296	0.02959
0.0231623795278	0.0460850283611	0.02345950129	0.02397
0.0195037311043	0.0389931600607	0.0198524835167	0.02322
0.0169146827448	0.0335281026399	0.0168725707329	0.01726
0.0125413801644	0.0249358420452	0.0128286870354	0.01378
0.0336853336321	0.0675887724487	0.0340887157945	0.02938
0.00737647738697	0.0145133565457	0.00747721357495	0.01798
0.0270749261504	0.0543530338617	0.0275193038864	0.04325
0.0229367539605	0.0455731635944	0.0230546541676	0.01236
0.0153923462771	0.0305684027608	0.0153765404848	0.01250
0.0276334577194	0.054889182742	0.0282327298443	0.02605
0.0167468084278	0.0333189664752	0.0166678952973	0.01067
0.0171655969335	0.0341504465775	0.0170995027749	0.02379
0.0108907676254	0.0220141884316	0.0110321539325	0.00502

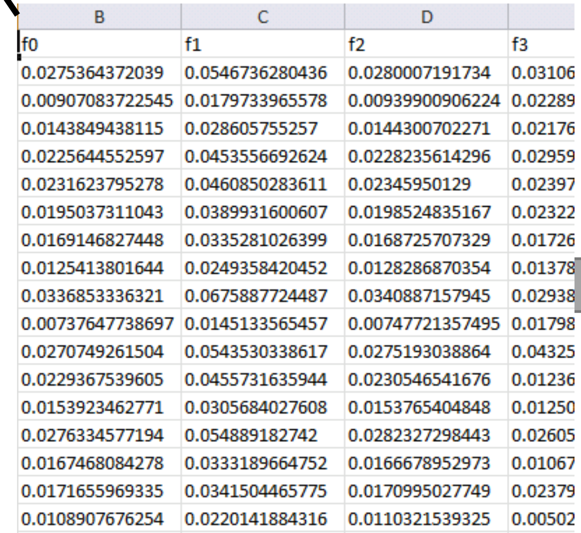


Machine Learning (supervised)

Test (unseen) data

Features

Samples

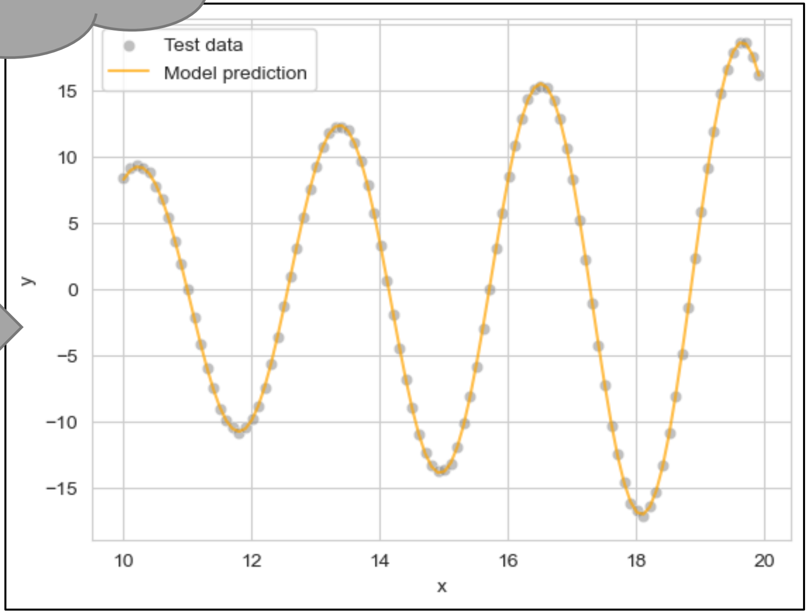


B	C	D	
f0	f1	f2	f3
0.0275364372039	0.0546736280436	0.0280007191734	0.03106
0.00907083722545	0.0179733965578	0.00939900906224	0.02289
0.0143849438115	0.028605755257	0.0144300702271	0.02176
0.0225644552597	0.0453556692624	0.0228235614296	0.02959
0.0231623795278	0.0460850283611	0.02345950129	0.02397
0.0195037311043	0.0389931600607	0.0198524835167	0.02322
0.0169146827448	0.0335281026399	0.0168725707329	0.01726
0.0125413801644	0.0249358420452	0.0128286870354	0.01378
0.0336853336321	0.0675887724487	0.0340887157945	0.02938
0.00737647738697	0.0145133565457	0.00747721357495	0.01798
0.0270749261504	0.0543530338617	0.0275193038864	0.04325
0.0229367539605	0.0455731635944	0.0230546541676	0.01236
0.0153923462771	0.0305684027608	0.0153765404848	0.01250
0.0276334577194	0.054889182742	0.0282327298443	0.02605
0.0167468084278	0.0333189664752	0.0166678952973	0.01067
0.0171655969335	0.0341504465775	0.0170995027749	0.02379
0.0108907676254	0.0220141884316	0.0110321539325	0.00502

ML algorithm
(model+optimizer)

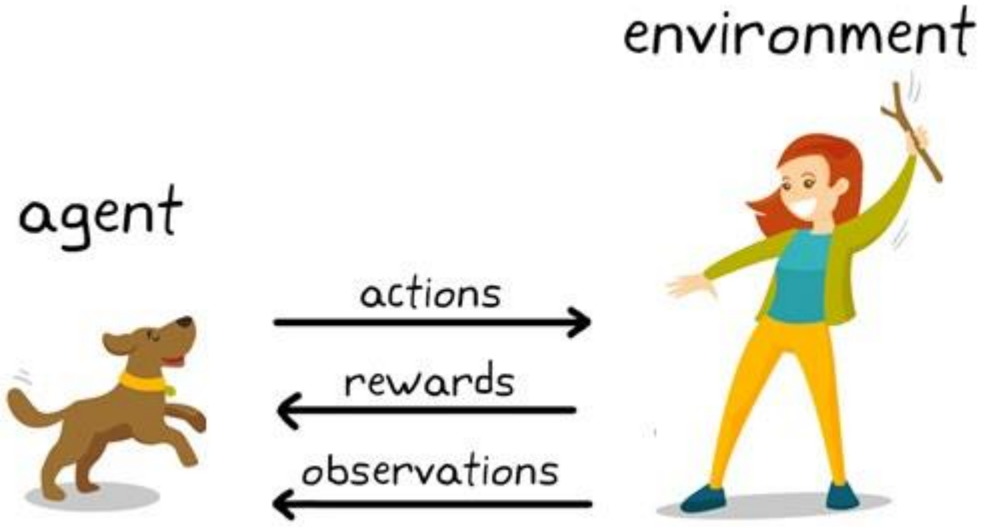
Internal parameters
(fixed)

Prediction...



Reinforcement Learning

- Similar to supervised ML, but not exactly
  - No value associated to a single decision
  - Reward is consequence of a *series* of decisions
  - Example: a chess game; should we trade a Queen for a Knight? Well, it depends on the **board state**
- Learn policy π
  - From **state** to **best action**
  - π can be approximated by a neural network



Reinforcement Learning

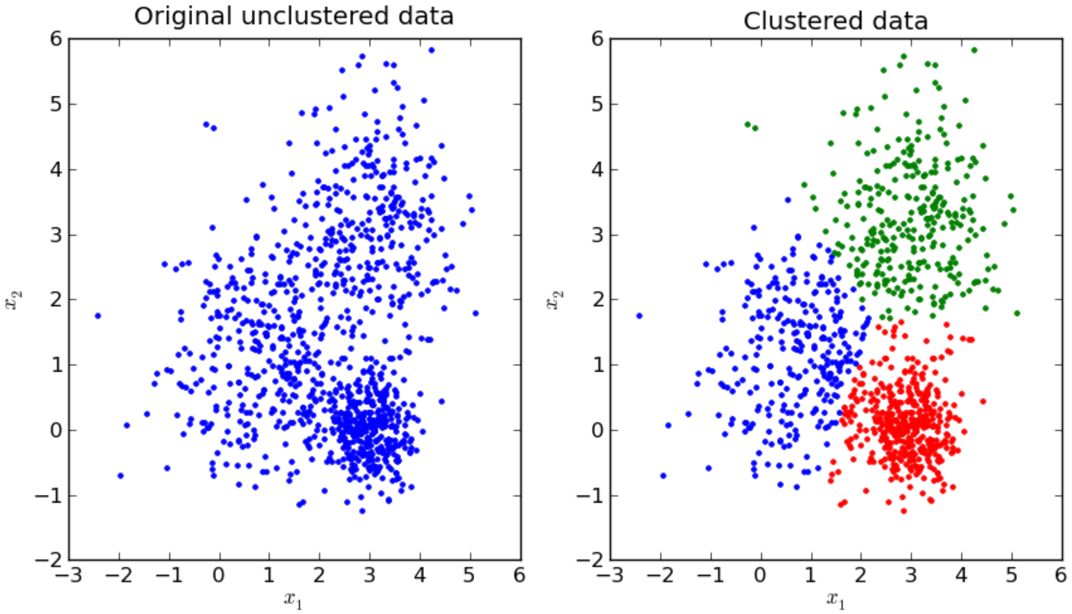
- Issues with the state space
  - Real-world applications have Vast search spaces
  - Even a game like chess has ~10^{20} possible board states
  - Impossible to explore exhaustively! (highest is checkers, 10^{10})
- Tricks to reduce states: exploit symmetries, remove useless…
- Game of Go (~10^{100} states) believed to be unapproachable
  - Estimated number of atoms in visible universe ~10^{80}
  - In 2016, world champion defeated! **Deep reinforcement learning**

Unsupervised Machine Learning

- What happens when we **do not have the ground truth**?

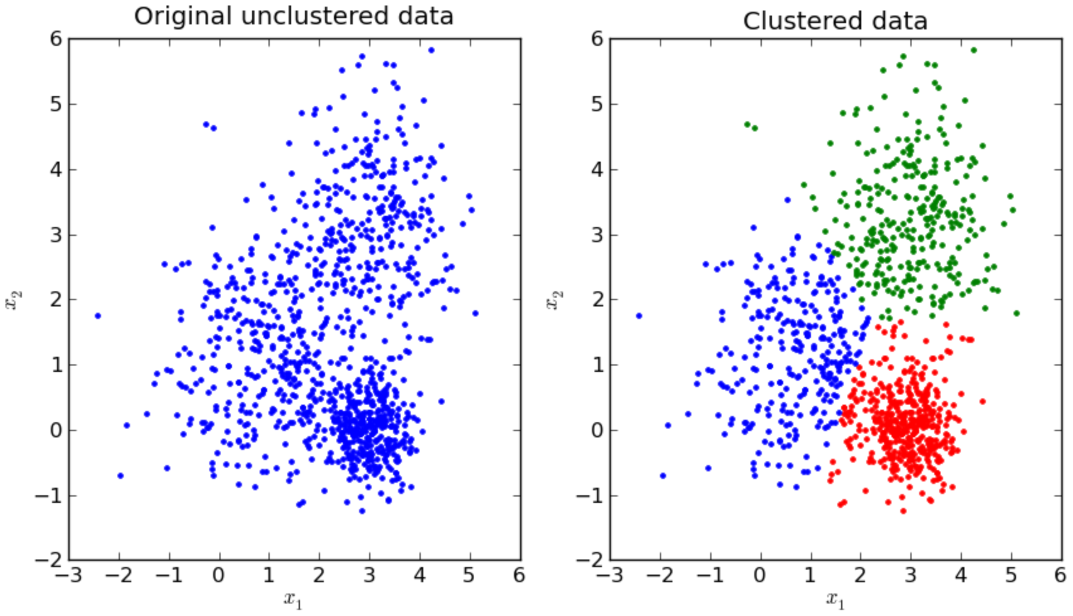
Unsupervised Machine Learning

- We suspect existence of regularities in the data
  - Find/create metric *correlated* with what we are looking for
  - Optimize the new metric, just as for classic ML
- Clustering
  - Find groups of data points that are *similar* to each other
  - What kind of metric could we imagine using here?



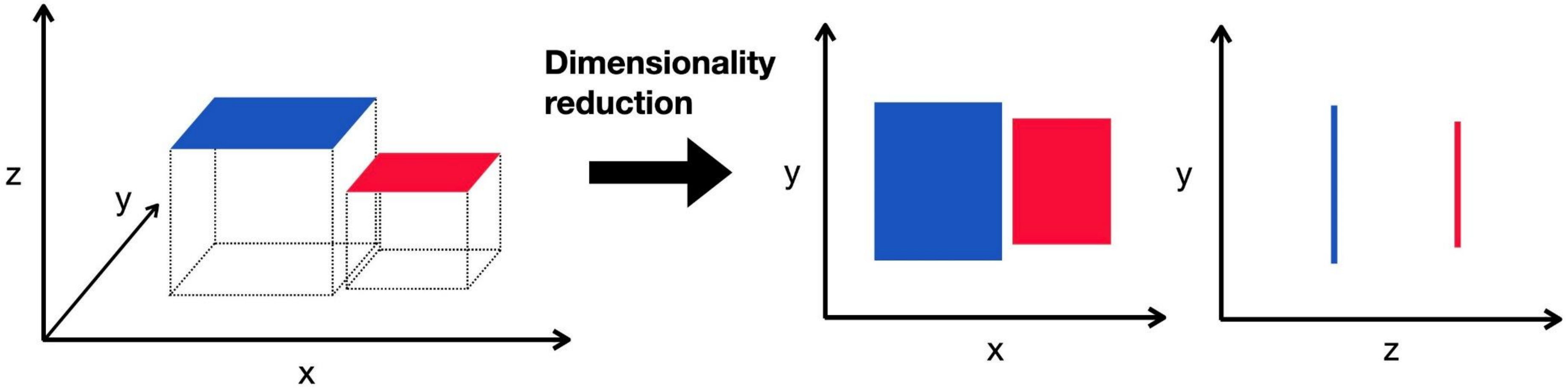
Unsupervised Machine Learning

- We suspect existence of regularities in the data
  - Find/create metric *correlated* with what we are looking for
  - Optimize the new metric, just as for classic ML
- Clustering
  - Find groups of data points that are *similar* to each other
  - **Minimize** *intra-group* distance, **maximize** *inter-group* distance



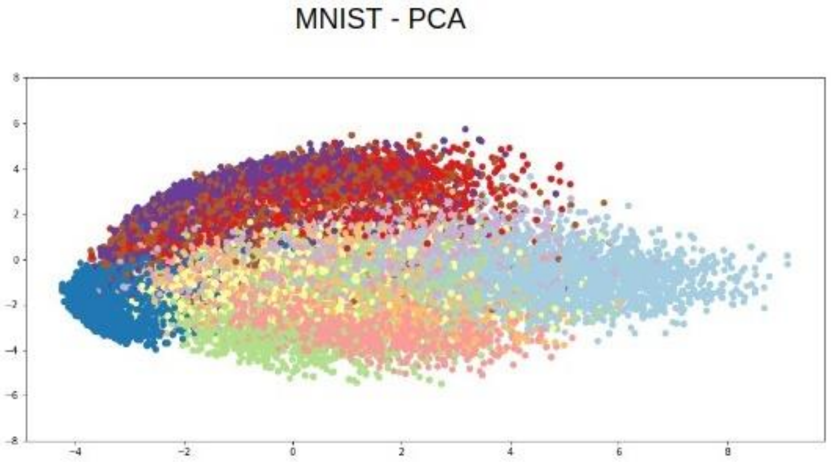
Unsupervised Machine Learning

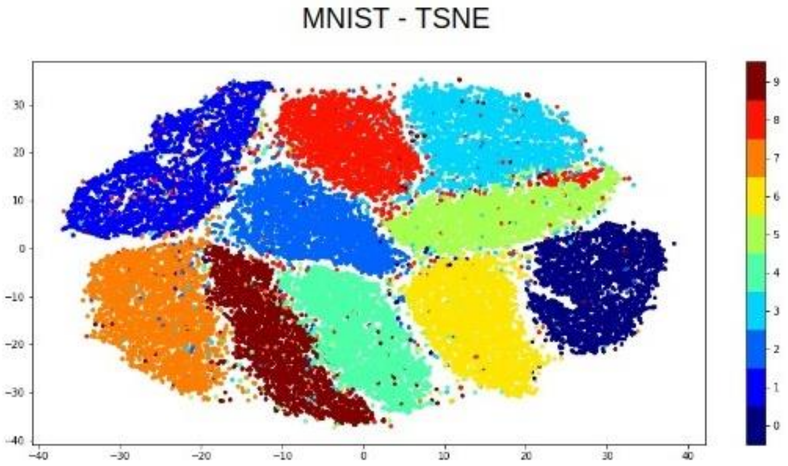
- Dimensionality reduction
  - Original feature space is high-dimensional
  - Cannot be easily visualized or understood by humans
  - Features could be useless or redundant



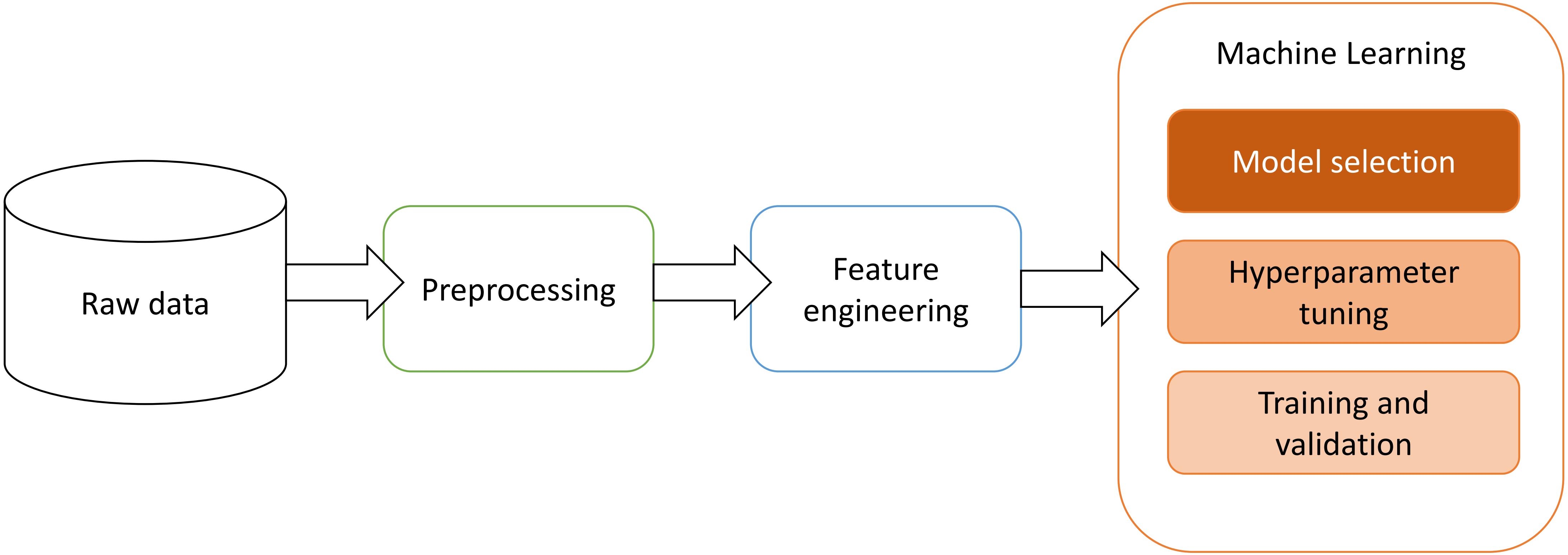
Unsupervised Machine Learning

- Create new dimensions as a combination of features
  - Principal Component Analysis, linear combination
  - Approaches creating non-linear combinations
    - t-distributed stochastic neighbour embedding (t-SNE)
    - Kernel PCA

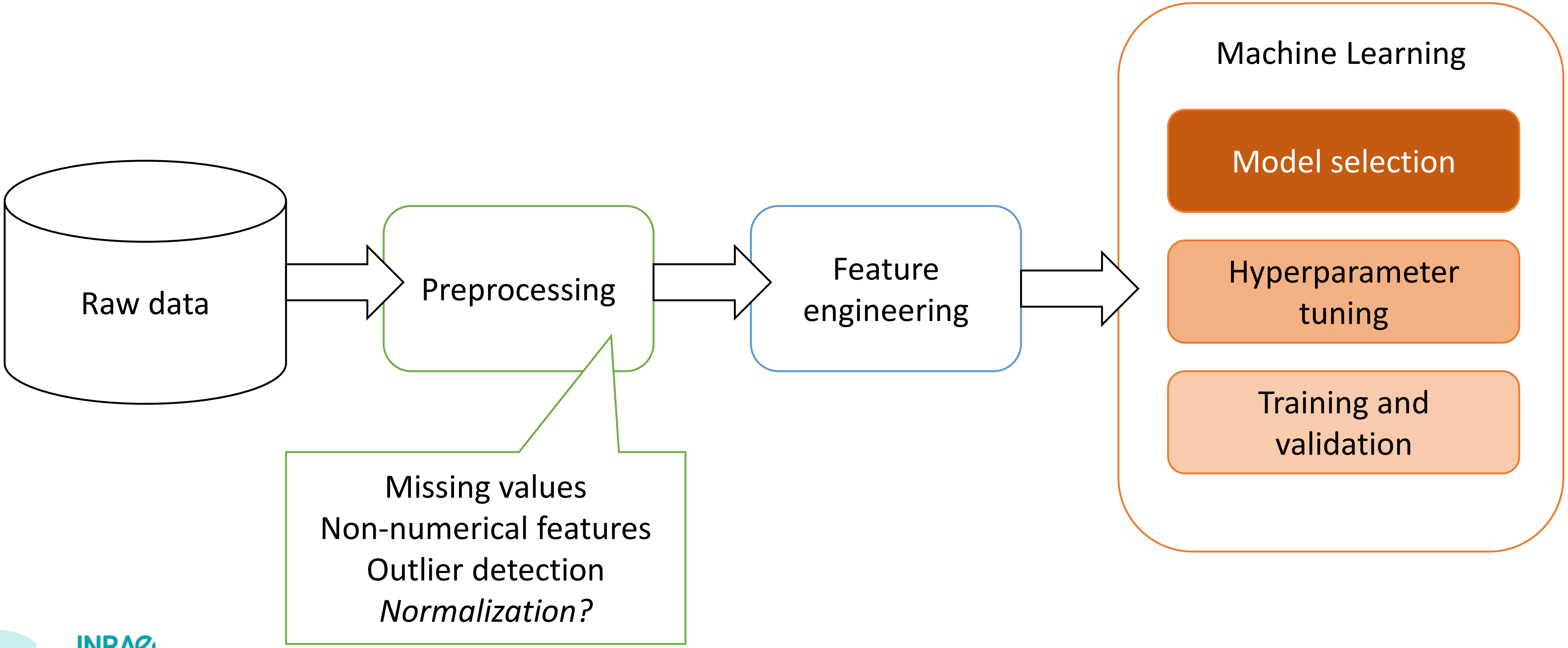

MNIST - PCA


MNIST - TSNE

Machine Learning pipeline



Machine Learning pipeline



Missing values

- ML algorithms cannot natively deal with missing data
- Trivial solutions
  - Remove samples with missing feature values
  - If a feature is often empty/NaN, remove whole feature
- Issue: this reduces the available data
- Can you think of any other solution?

Missing values

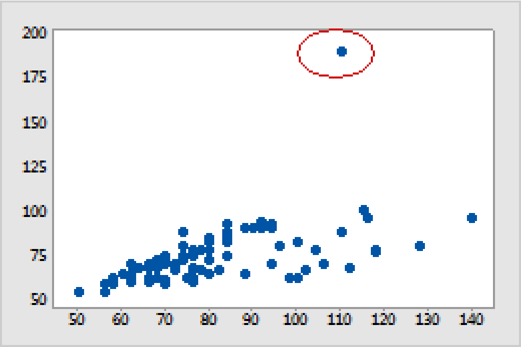
- **Imputation**
  - Replace the missing value with *another value*; but which one?
  - Zero, -1, (…)
  - Mean/median value of the feature over all samples/same class
  - Expert judgment (if not too many missing values)
- Machine learning for imputation
  - Train a ML model to predict the value of the feature
  - Example: KNN imputation
- (IMHO) If possible, avoid imputation or use experts

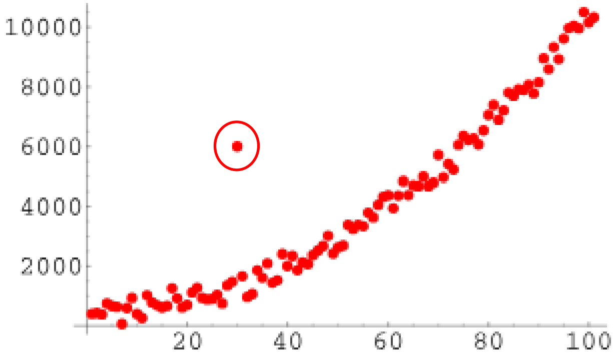
Non-numerical features

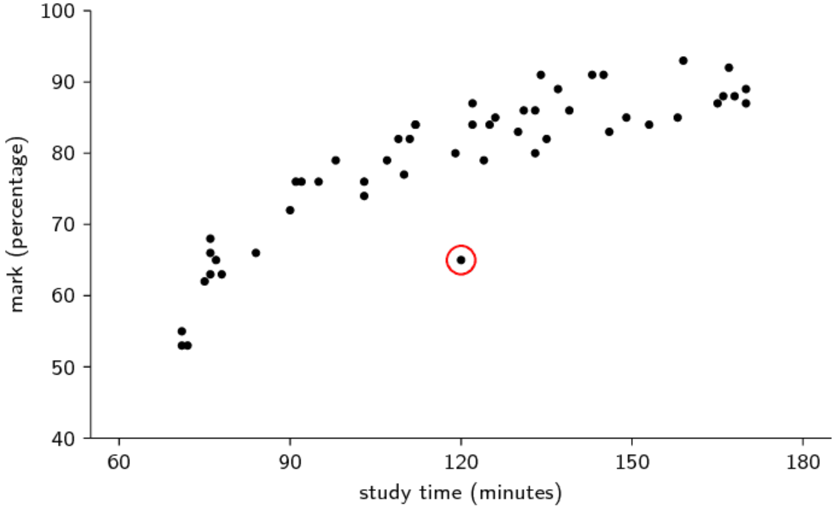
- Categorical features to numbers?
- If ordered (“high”/“medium”/“low”), to **integers** (2/1/0)
- If not ordered (“red”/“blue”/“green”), **one-hot encoding**
  - Create additional binary (0/1) features, equal to number of values of categorical feature
  - Set binary feature to ‘1’ and others to ‘0’ to represent values
  - E.g. red=100, blue=010, green=001
- Utils in Pandas that already take care of (most of) this

Outlier detection

- What is an *outlier*?







Outlier detection

- While the idea is intuitive, its application is difficult
  - Sometimes outliers are errors in data collection…
  - …but sometimes they are representative of the phenomenon
  - “Out of Distribution”, but can we identify the distribution?
- Machine learning methods
  - Isolation Forest
  - Local Outlier Factor
  - Problem sometimes called “Novelty Detection”
- (IMHO) Expert knowledge or avoid removing outliers

Normalization?

- Several algorithms need feature values to be in (0,1) or (-1,1)
  - Optimization algorithms in the ML approach work better in range
  - Or other possible numerical issues (e.g. values too big)
- Library functions automatically perform normalization
- Example: rescale to zero mean and unit variance

$$z_i = \frac{(x_i - \mu)}{\sigma}$$

- So, just apply normalization to the data…right?

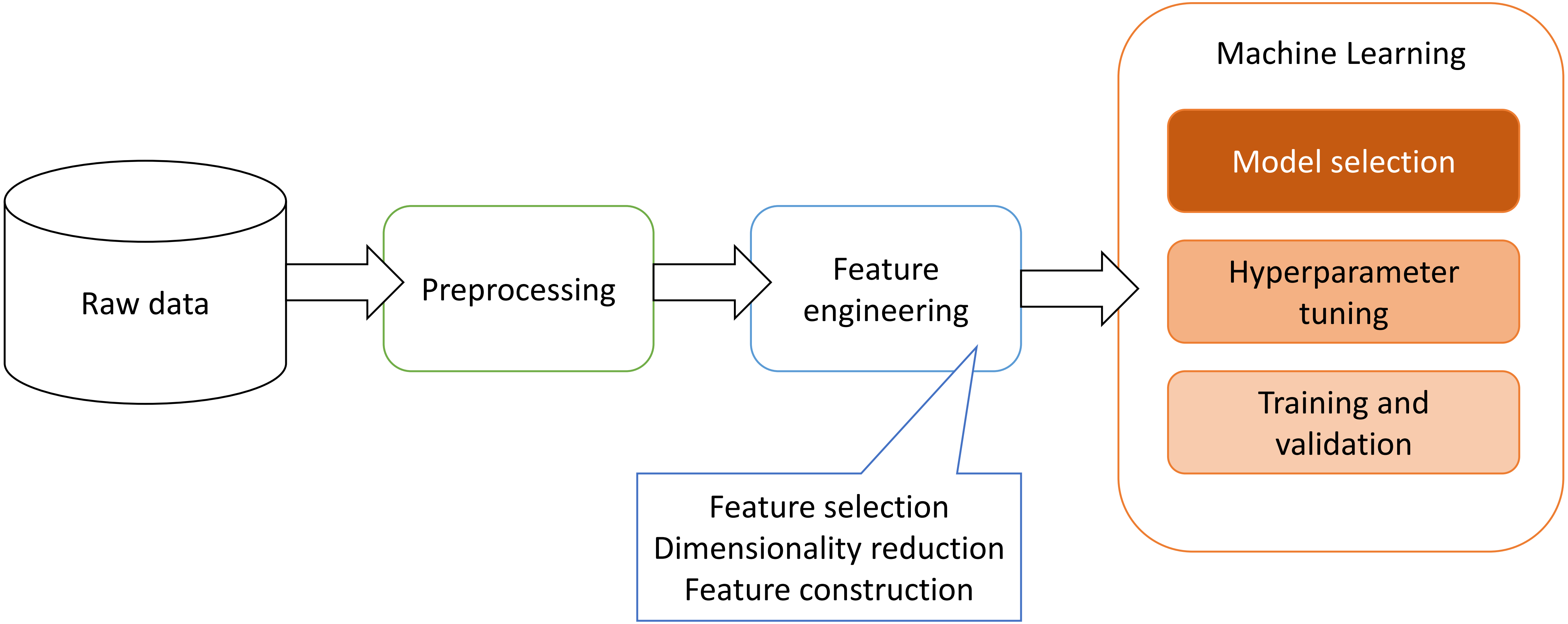
Normalization

- No! Normalization has *parameters learned on data!*

$$z_i = \frac{(x_i - \mu)}{\sigma}$$

- Learning parameters and applying to all data is **overfitting**
- The impact is usually not huge, but it can be important
- Choose normalization algorithm
  - Apply it later, during the training and validation step
  - Learn normalization from training set, apply to test set
  - …unless **you already know min/max** and want to just rescale (0,1)

Machine Learning pipeline



Feature selection

- Why reduce the number of features in the problem?

Feature selection

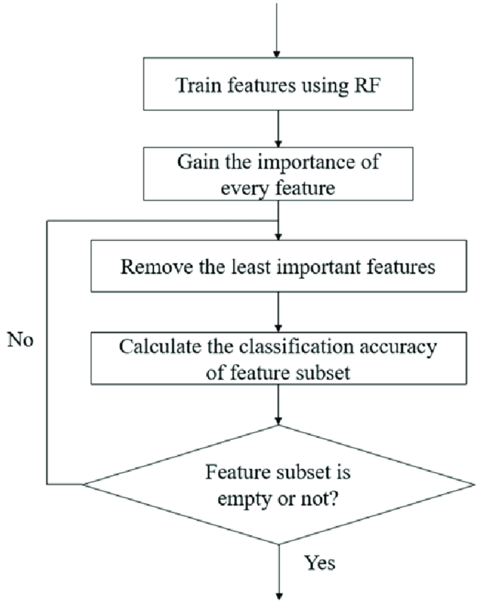
- Human-readability: make sense of 10 features, not 100
- Improve performance
  - Difficulty of the task is usually correlated with number of features
  - Sometimes, ill-posed problems (more features than samples)
  - Reducing features might improve behavior of ML algorithm

Feature selection

- Remove all *useless* or *redundant* features
  - Useful: features (strongly) correlated with the target
  - Redundant: features (strongly) correlated with other features
- Filter methods
  - Analyze simple univariate correlations
  - For example, mutual information or covariance
  - Strong hypothesis: contribution of features is separable
  - Example: SelectKBest function in scikit-learn
- Embedded methods (as part of a ML algorithm, LASSO)

Feature selection

- Wrapper methods
  - Use a ML algorithm in a loop, evaluate its performance
  - Search space of all possible feature subsets
  - Recursive Feature Elimination (RFE), Permutation Importance



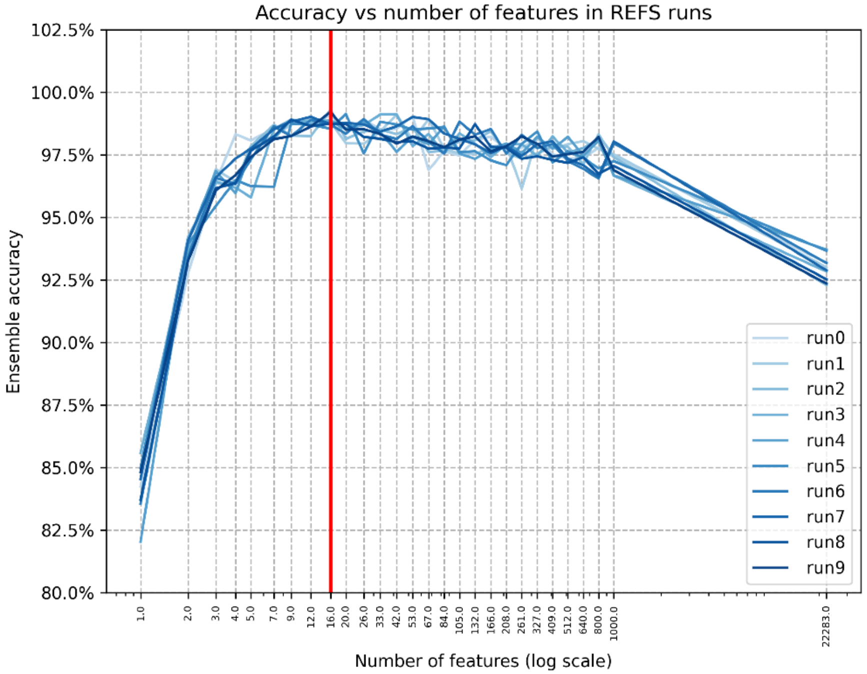
Train features using RF

Gain the importance of every feature

Remove the least important features

Calculate the classification accuracy of feature subset

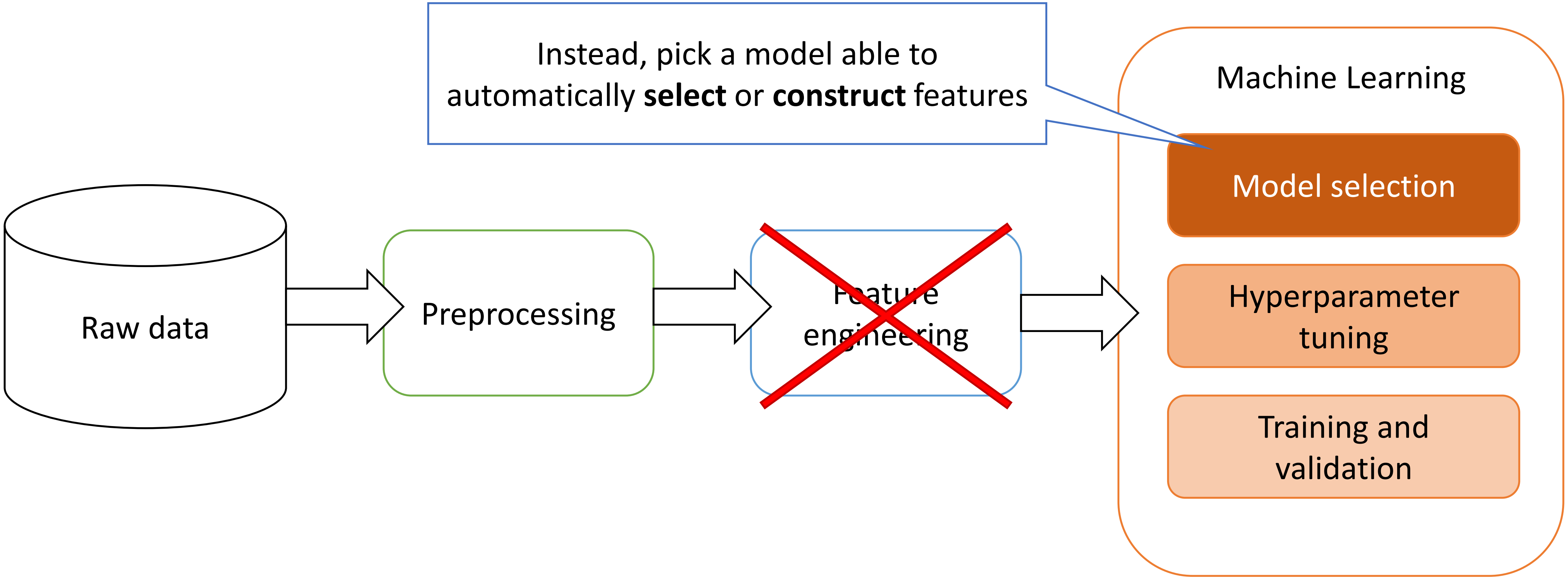
Feature subset is empty or not?



Feature construction

- Typical issue with **relational data**
- Raw features of the problem are not really relevant
- What is important is a *combination* of the raw features
  - Example: single pixels in an image are not very important
  - What matters are recurring *patterns of pixels*
- A fundamental problem for lots of practical applications
- Features hand-crafted by human experts of the problem
- Limited success, especially with images

Deep Learning pipeline…?



Instead, pick a model able to automatically **select** or **construct** features

Raw data

Preprocessing

~~Feature engineering~~

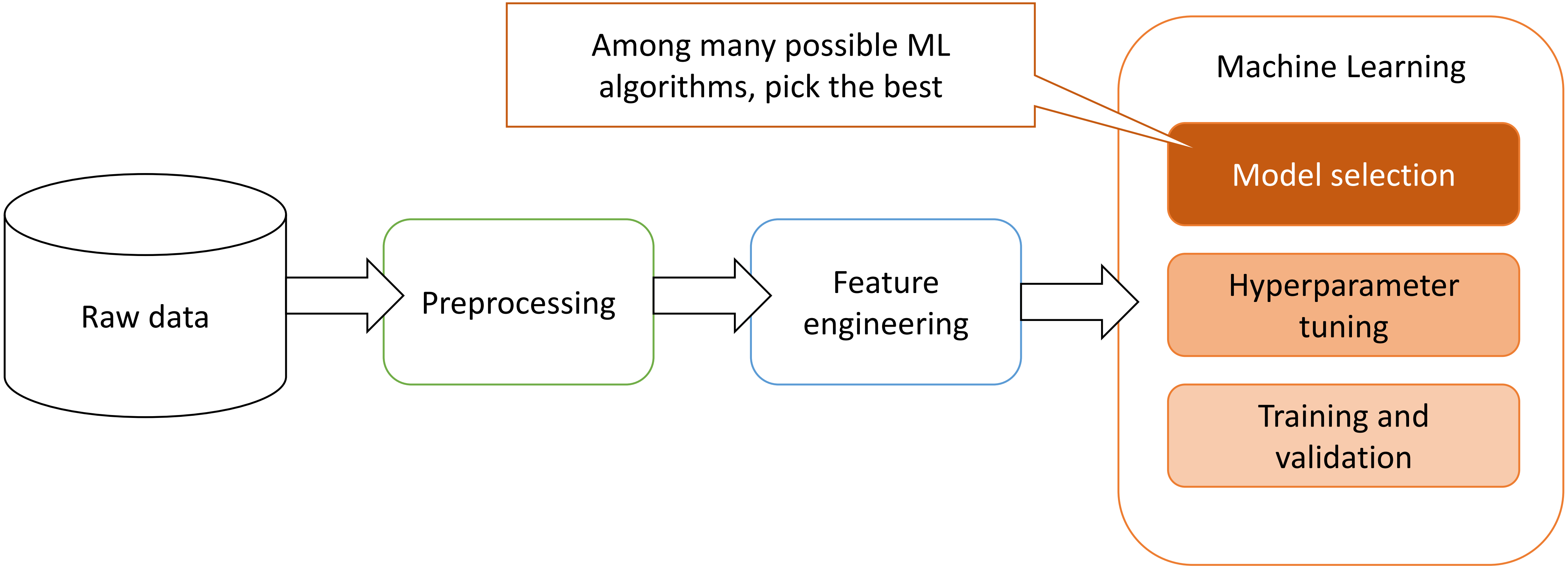
Machine Learning

Model selection

Hyperparameter tuning

Training and validation

Machine Learning pipeline



Among many possible ML algorithms, pick the best

Raw data

Preprocessing

Feature engineering

Machine Learning

Model selection

Hyperparameter tuning

Training and validation

Model selection

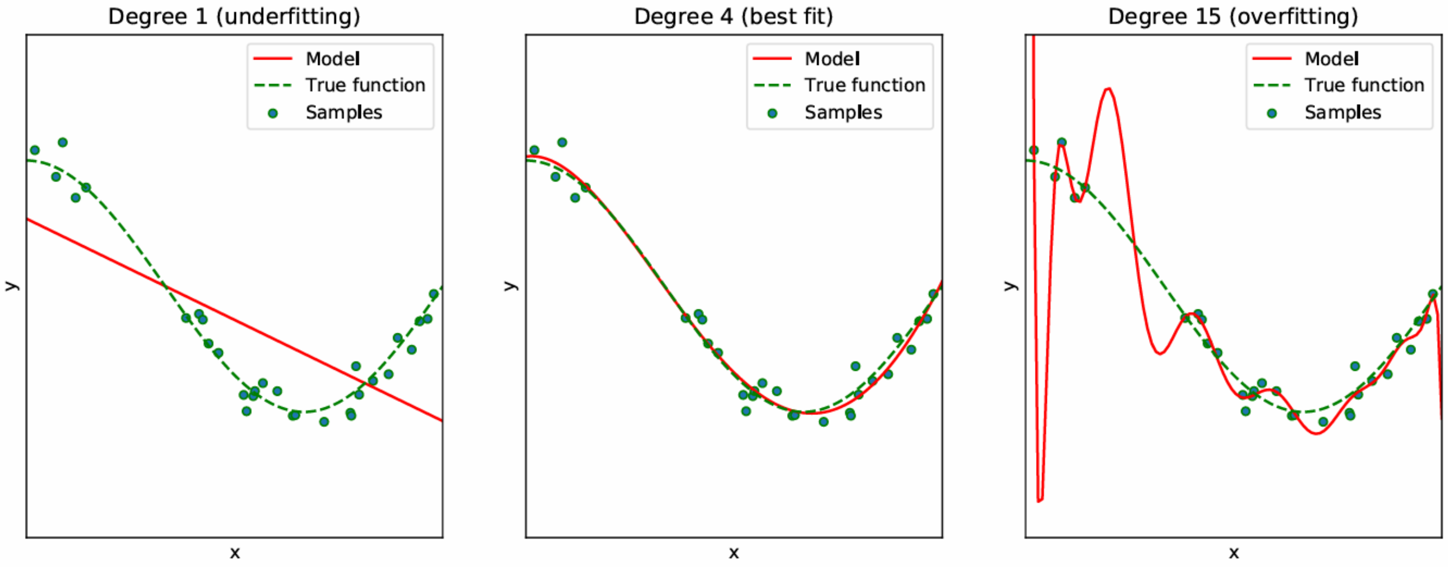
- Model/ML algorithm with **best performance** on training data
- What if multiple algorithms have similar performance?

Model selection

- Model/ML algorithm with **best performance** on training data
- Among models with similar performance, **lowest capacity**

Capacity?

- The most complex function a ML model can approximate
  - A **linear model** cannot fit a *quadratic function* well
  - A **quadratic model** cannot fit a *cubic function* well (…and so on)
- However, larger capacities are also linked to **overfitting**



Capacity and overfitting

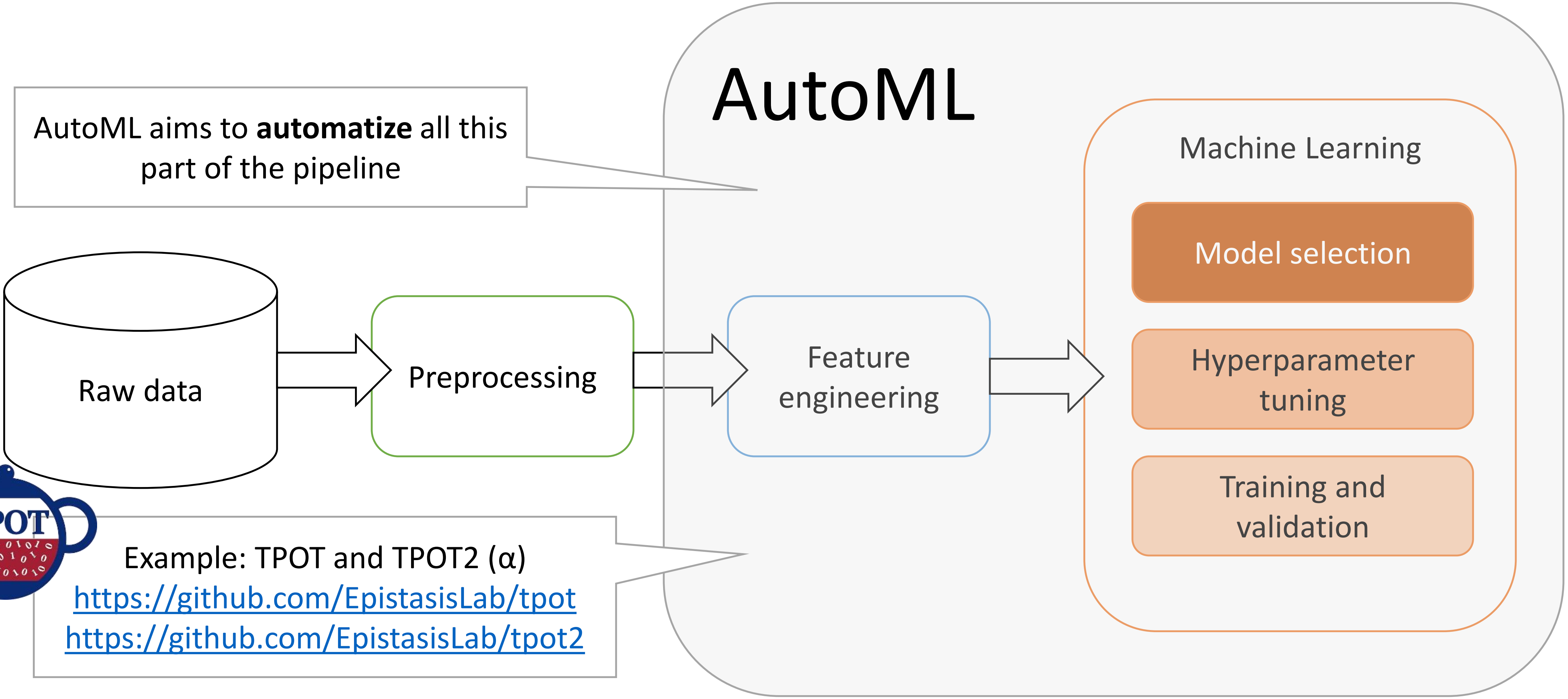
- Overfitting
  - The performance on training does not generalize to unseen data
  - Model captured correlations that only exist in training (e.g. noise)
- High capacity fits well, may lead to overfitting
- Low capacity might underfit, performance more predictable
- Know optimal capacity needed for a target data set?
  - No.
- Estimate and compare capacity of different algorithms?
  - Also no. (*maybe* upper bound, **Vapnik-Chervonenkis dimension**)

Model selection in practice

- Try as many different algorithms as you can
  - Evaluate algorithms in a cross-validation on training data
  - Check mean and standard deviation of performance
  - Best mean, lowest stdev (more reliable)
- Alternative: AutoML
  - Search a Vast space of possible algorithms (and configurations)
  - Still extremely slow and computationally expensive
  - Growing research interest

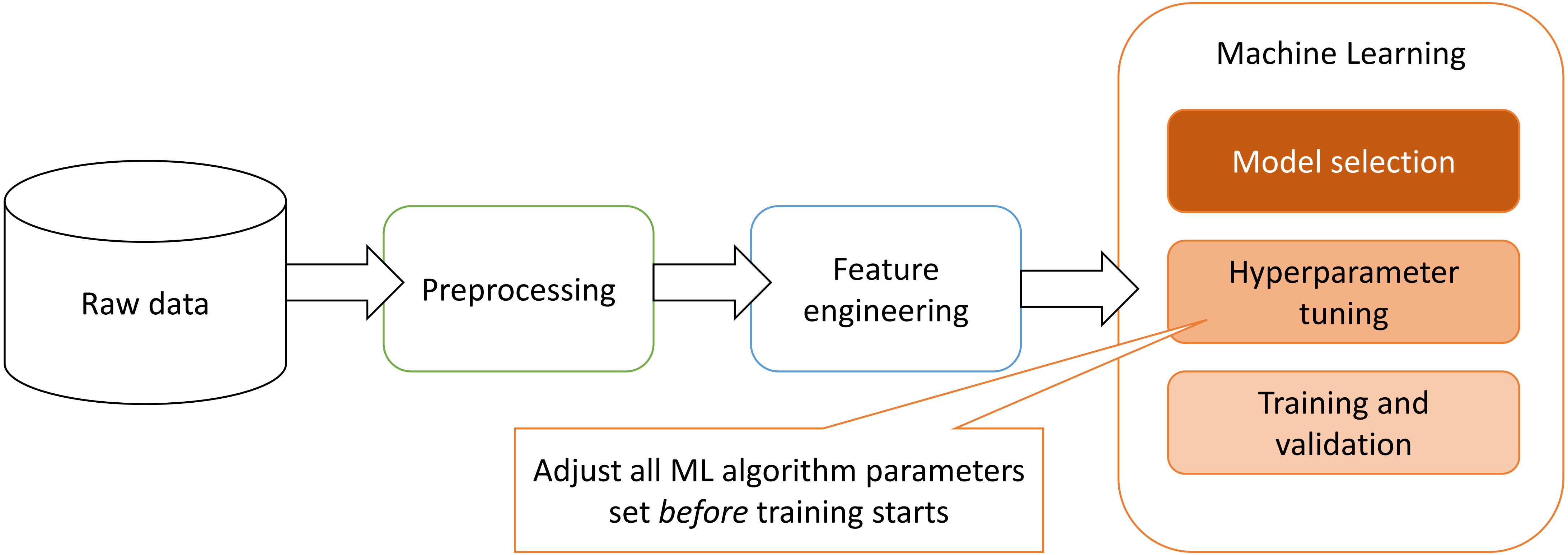
AutoML?

AutoML aims to **automatize** all this part of the pipeline



Example: TPOT and TPOT2 (α)
https://github.com/EpistasisLab/tpot
https://github.com/EpistasisLab/tpot2

Machine Learning pipeline



Raw data

Preprocessing

Feature engineering

Machine Learning

Model selection

Hyperparameter tuning

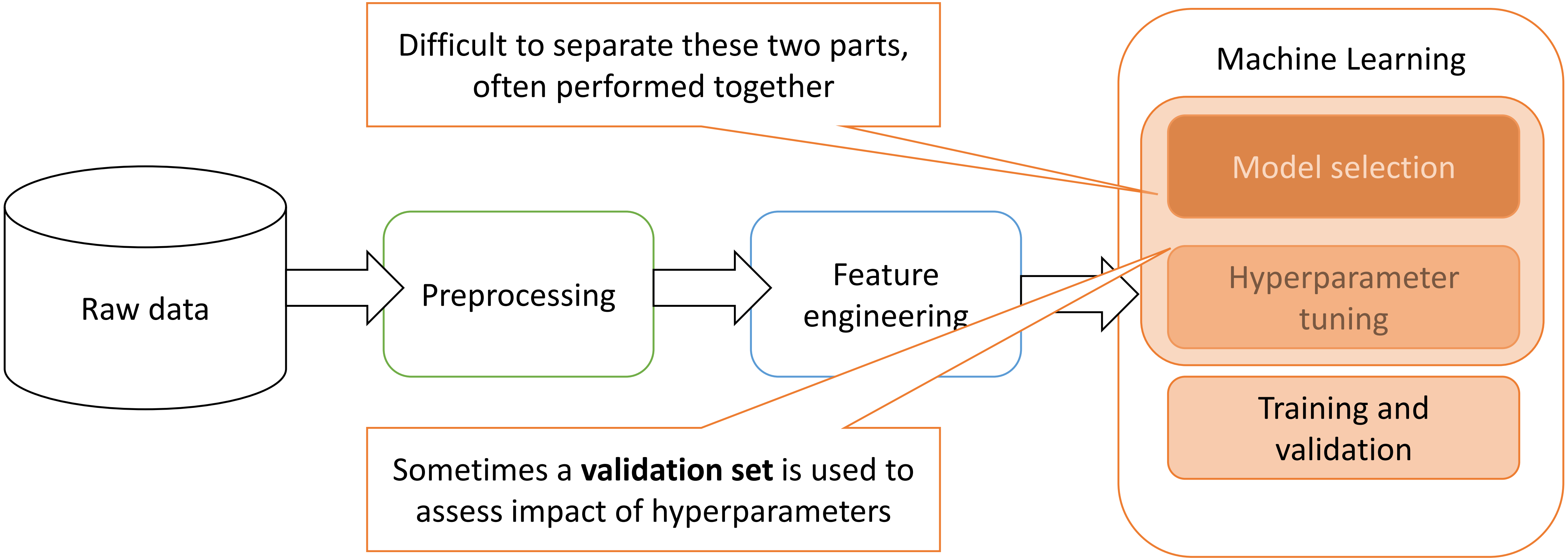
Training and validation

Adjust all ML algorithm parameters set *before* training starts

Hyperparameter tuning

- Hyperparameters?
  - Parameters are all numbers “inside” a ML model
  - Parameters are adjusted during the learning/training process
  - *Hyper*-parameters are set *before* training starts
- Examples of hyperparameters
  - In a Random Forest, number of trees, depth of trees, criterion for creating splits in trees
  - In a Neural Network, number of layers, number of neurons per layer, activation function of neurons, optimization algorithm, …
- Hyperparameters **impact capacity**

Machine Learning pipeline



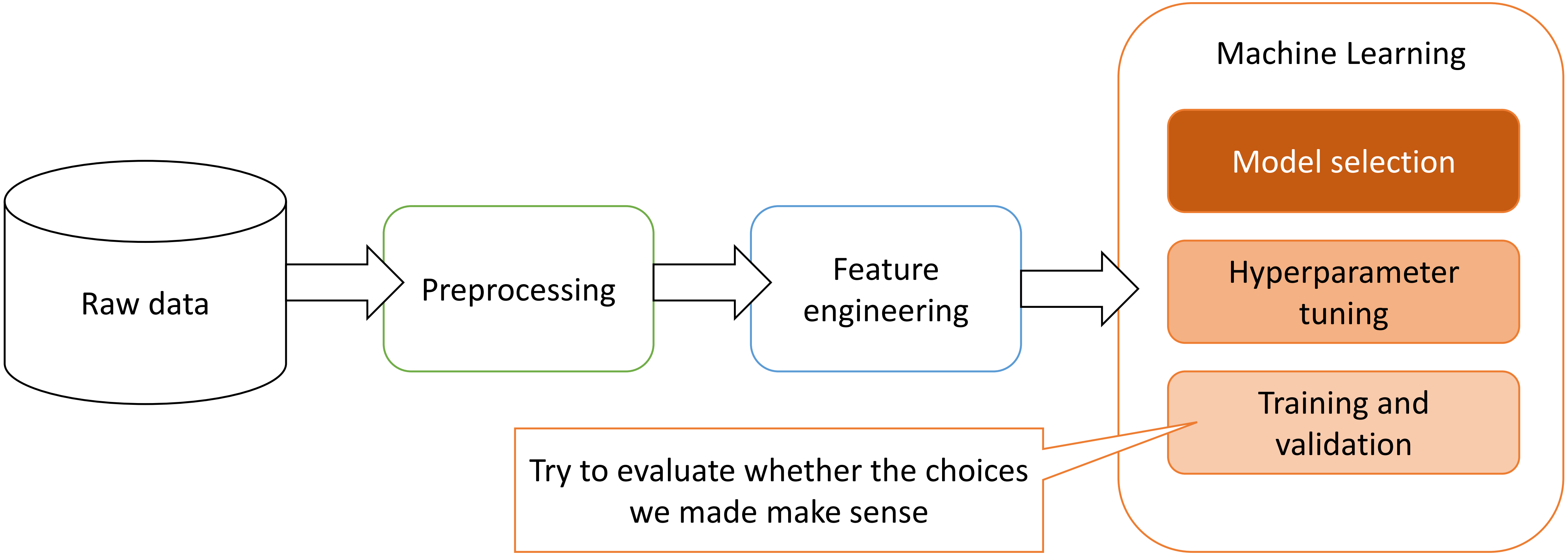
Difficult to separate these two parts, often performed together

Sometimes a **validation set** is used to assess impact of hyperparameters

Hyperparameter tuning

- Hyperparameters have little impact* on performance
    - *not always true, but true for lots of practical applications
    - So, in principle you can pick model first, and then hyperparameters
- However, the above is **absolutely not true** for Deep Learning
    - DL has **way more hyperparameters** than other methods
    - Hyperparameters literally *make or break* a DL network
    - Changing the *learning rate*, for example, has HUGE IMPACT
    - Subset of AutoML *exclusively dedicated* to DL
    - Neuro-evolution, Neural Architecture Search

Machine Learning pipeline



Raw data

Preprocessing

Feature engineering

Machine Learning

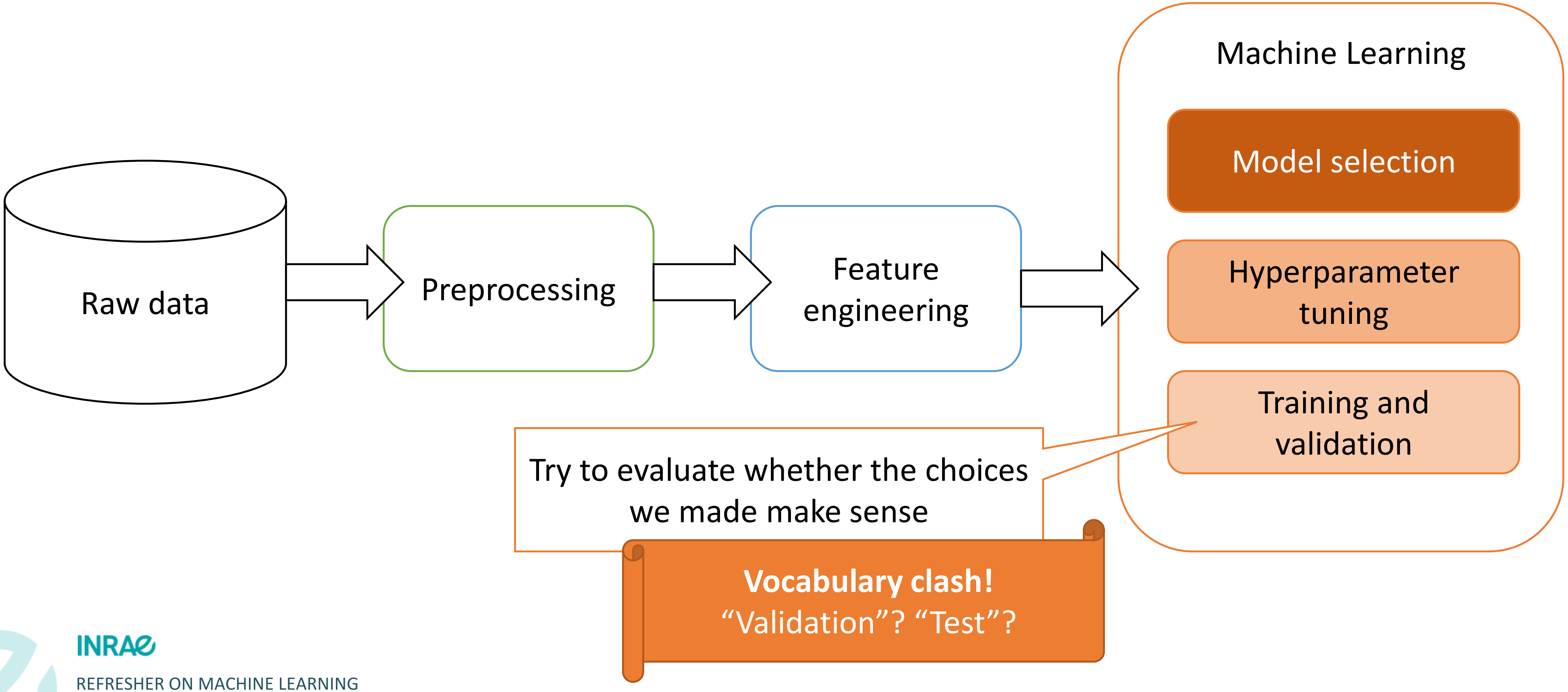
Model selection

Hyperparameter tuning

Training and validation

Try to evaluate whether the choices we made make sense

Machine Learning pipeline



Try to evaluate whether the choices we made make sense

Vocabulary clash!
“Validation”? “Test”?

Why do we need test/validation?

- Overfitting is the **final boss**
  - We want to be sure that model generalizes
  - Test data: unseen (during training)
  - Risk that the model captures unique properties of the training data…
  - …that only exist for that training set!
- How to **evaluate overfitting**?

AI-generated image, prompt "The concept of overfitting in machine learning as the final boss monster in a videogame"

Why do we need test/validation?

- In theory, we would need *unseen data* (that we don't have)
- Train/test split: hide part of the available data, use it for test
- Even better: **k-fold cross-validation** (k=5 or 10)
  - Divide data into k parts (folds), then iterate k times
  - Each time, use k-1 folds for training; one fold for testing
  - Obtain an **average** and a **standard deviation** of performance
- Large stdev usually indicates issues: outliers?

Algorithms work inside a computer

- How are numerical values represented inside a computer?

Algorithms work inside a computer

- How are numerical values represented inside a computer?

1	1	0	1	0	0	0	1	…

$\underbrace{\hspace{20em}}$

Sequence of bits (binary values)
of **fixed length** (8, 16, 32…)

Algorithms work inside a computer

- *Limit* to minimum and maximum representable
- *Limit* to the smallest detectable difference (**machine epsilon**)
- Precision can be increased, at the **expense of memory**

(Pseudo-)Random number generation

- There is no *true* **random number generation** in a computer
  - Algorithms generate sequences of pseudo-random numbers
  - But after a (long) time, they start repeating
- Entire field of research on PRNG
- PRNG algorithm initialized with a certain value (*seed*)
  - If no value is specified, system time converted to integer
  - If the seed is the same, the sequence will be the same
- **Set and store the random seed** to reproduce experiments

Vocabulary

- **ML algorithm**: model type + optimization algorithm
- **Model/predictor**: one trained ML algorithm
- **Model parameters**
  - Values (numerical, categorical, …) *inside* the model
  - Optimized (e.g. change values) during training process
- **Samples**: rows of the dataset
- **Features**: columns of the dataset
- **Target(s)**: feature(s) we are interested in predicting

Vocabulary

- **Training data**: data from which we want to learn
- **Test data**: unseen data, kept aside to assess *generalization*
- **Validation data**: used during training, not for training (!)
- **Training/Fit**: optimize parameter values to fit training data
- **Cross-validation**: iterative performance where data is split into different training/test sets, and model re-trained

Vocabulary

- **Model hyperparameters**
  - Choices/parameters *outside* the model
  - Usually user-defined *before* training process starts
- **Capacity** (loose definition)
  - Maximum order of function that can be approximated by model
  - The more parameters, the more capacity
- **Bias**: source of errors, not enough capacity (underfitting)
- **Variance**: sensitivity to small variations in training data, too much capacity (overfitting)

Questions?

Bibliography

- James et al. 2023. *An Introduction to Statistical Learning with Applications in Python*

Images and videos: unless otherwise stated, I stole them from the Internet. I hope they are not copyrighted, or that their use falls under the Fair Use clause, and if not, I am sorry. Please don't sue me.