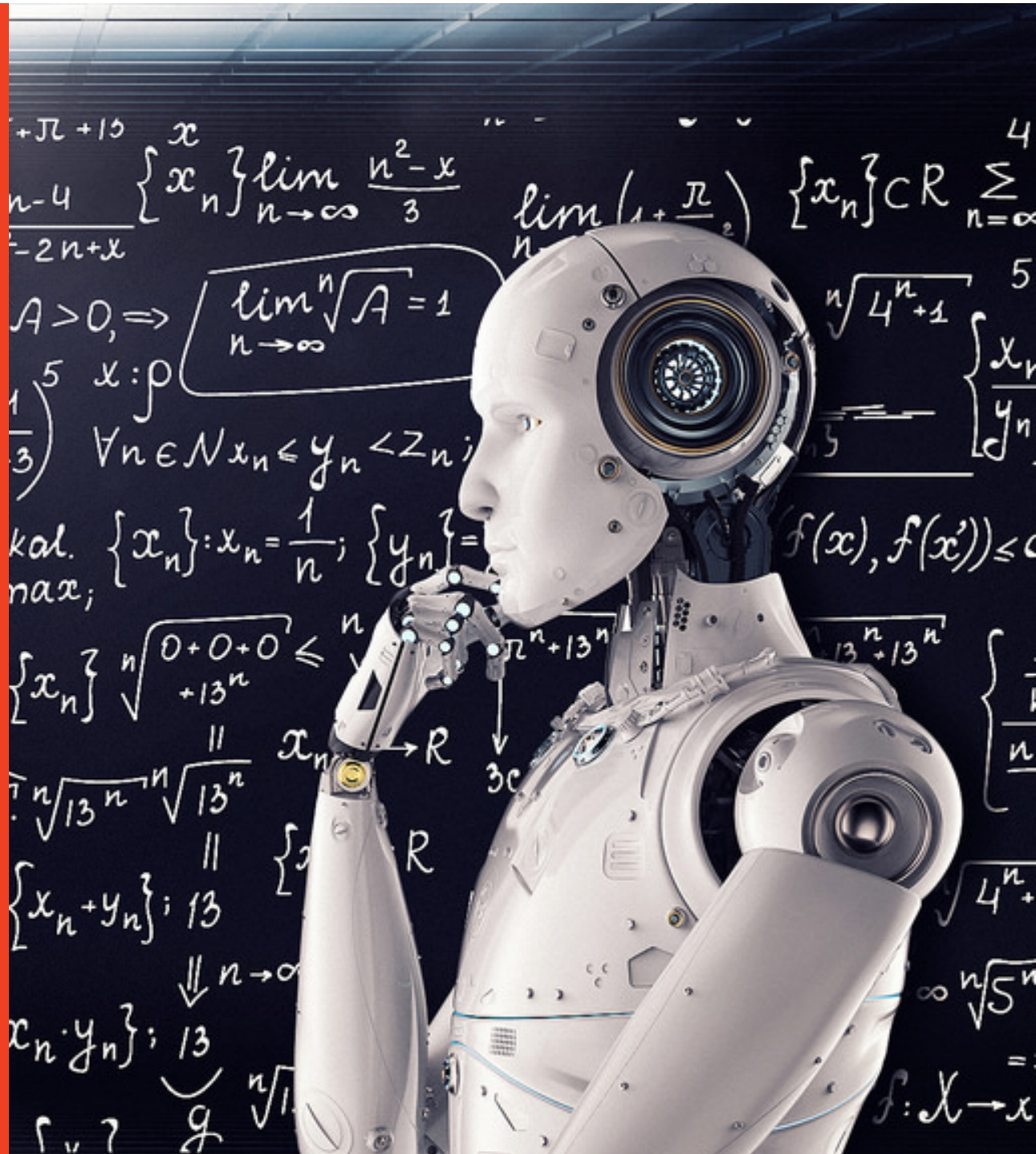
ANCHORMEN
data activators

PYTHON MACHINE LEARNING BASICS

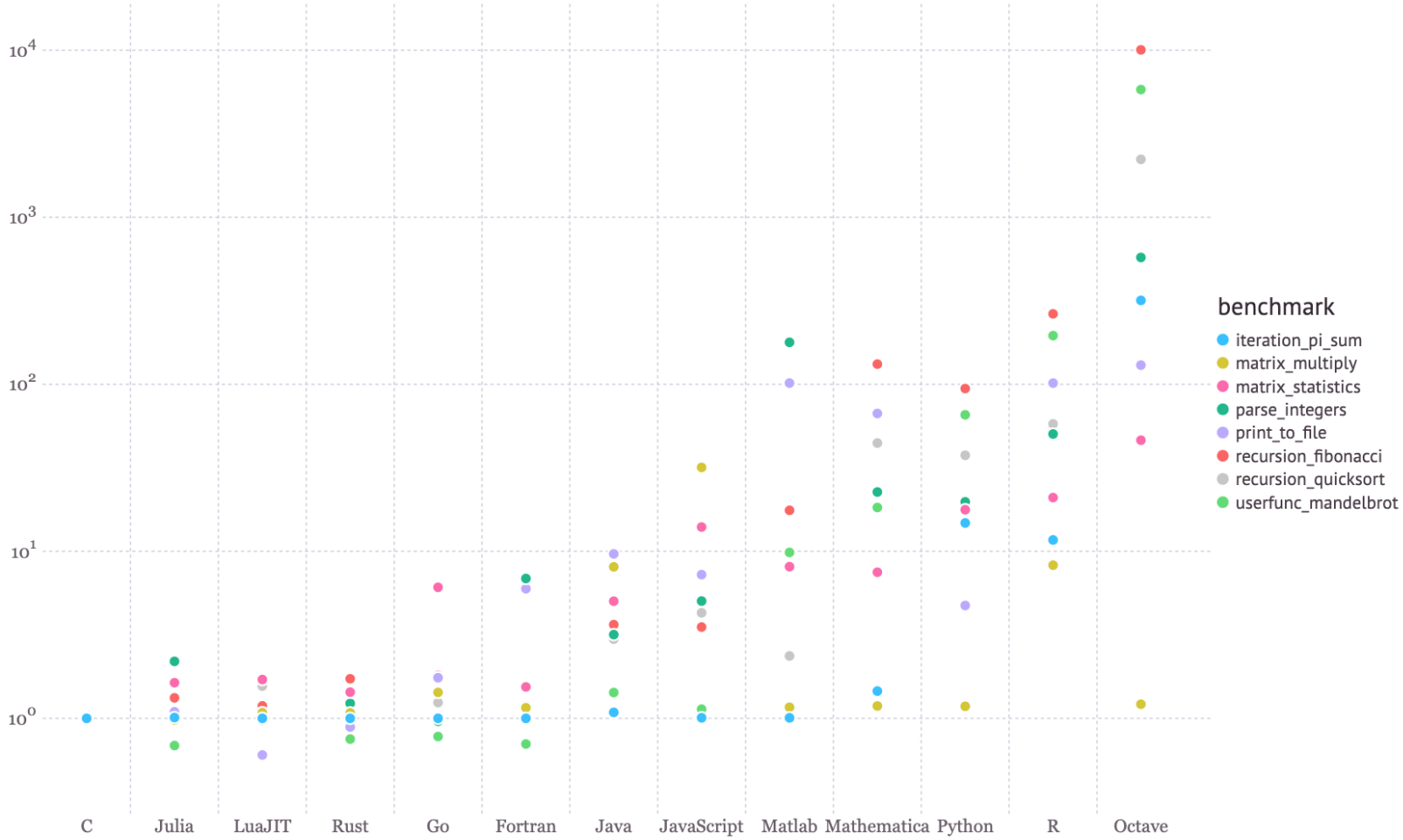


VECTORISATION

Python can be really slow.

Unless we can use the parts of python that are implemented with a C backend, like numpy.

This means we want to use things like matrix multiplication, if possible.



MATRIX MULTIPLICATION

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix}$$

MATRIX MULTIPLICATION

$$
\begin{aligned}
A\mathbf{x} &= \begin{bmatrix} 1 & -1 & 2 \\ 0 & -3 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} 2 \cdot 1 - 1 \cdot 1 + 0 \cdot 2 \\ 2 \cdot 0 - 1 \cdot 3 + 0 \cdot 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ -3 \end{bmatrix}.
\end{aligned}
$$

LAB 1

Let’s test this for ourselves and compare the speed differences.

DATA PROBLEMS

- Insufficient quantity
- Nonrepresentative data (sampling noise, sample bias)
- Poor quality data
- Irrelevant features
- Overfitting / underfitting

DATA PROBLEMS

Insufficient quantity

It is really common to have a client that has timeseries data (e.g. sales of a product) on a monthly or weekly level, 3 years backwards. This results in 3 * 12, or sometimes 3 * 52 , datapoints. That is not a lot for machine learning.

For medical data, it is really common to have something along the lines of 100 patients. Maybe 200, or 400. But that's a lot, already. Again, this is in most cases not enough for machine learning, even though this also depends on the type of data.

Why do you think that deep learning needs so much data?

How do you determine if a model is simple?

Solution: keep your models as simple as possible.

DATA PROBLEMS

Nonrepresentative data (sampling noise, sample bias)

Noise will always be something to consider. This can be combated with larger datasets.

However, sample bias is different. Here, a critical review of *how* the data is acquired is necessary.

This can be very tricky…

Solution: think through all possible biases.

DATA PROBLEMS

Nonrepresentative data (sampling noise, sample bias)

COMPAS algorithm, used in US to predict recidivism for criminals

Algorithm turned out to be racist: Blacks where misclassified twice as much

Cause: a questionnaire was used, in which the bias of society itself was reflected. E.g. both blacks and whites smoke as much marihuana (as can be measured in sewer water), but blacks are 10 times as likely to get a criminal record for this.

Sources:
https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing
https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm
https://github.com/propublica/compas-analysis

DATA PROBLEMS

Poor quality data

Examples:

- Finding a correlation between weatherdata and data, but having to predict 8 weeks ahead. How do you get realistic weather data for the future? You don't.
- Human influences are more common then you might expect. E.g. the amount of tires, ordered by car manufacturers is highly influenced by all sorts of ad hoc decisions (e.g. planning problems, switching weeks). As long as you are unaware of the reasons behind these changes, these changes should be look at as noise. Be aware that a human, that actually knows what the reason behind these changes is, might not be aware how polluted the data is. To him, this 'just makes sense because reason xyz'.

Solutions:

(i) Be critical in the features you use in your model. More features is not always better.

(ii) Talk to the domain experts about the data. Standardize the gathering of data and solve what is left with data preprocessing and removal of outliers.

DATA PROBLEMS

Irrelevant features

Sometimes you have hundreds of features (e.g. 10.000 genes to predict 3 types of breast cancer). A huge percentage will be irrelevant, but how do you find out?

Solutions: first prune your features, then reduce dimensionality and regularize your model.

PS: this topic deserves more attention, which it will get in future lessons.

DATA PROBLEMS

Overfitting / underfitting

This mainly means: using a too complex or too simple model.

But also: your feature generation might be too complex.

E.g. : Having decomposed timeseries (24 hours) for a normal day, winter day, summer day, and in addition for 12 months and 7 weekdays gives you 91 features.

While there will be a lot information in this feature space, trying to directly fit a K-means clustering on these 91 features is not a good idea (Why not? Hint: can you give an estimate of how big the feature space is, even if every feature has only two relevant values?).

Solution: keep your models as simple as possible, both on the side of the model itself as on the feature side.

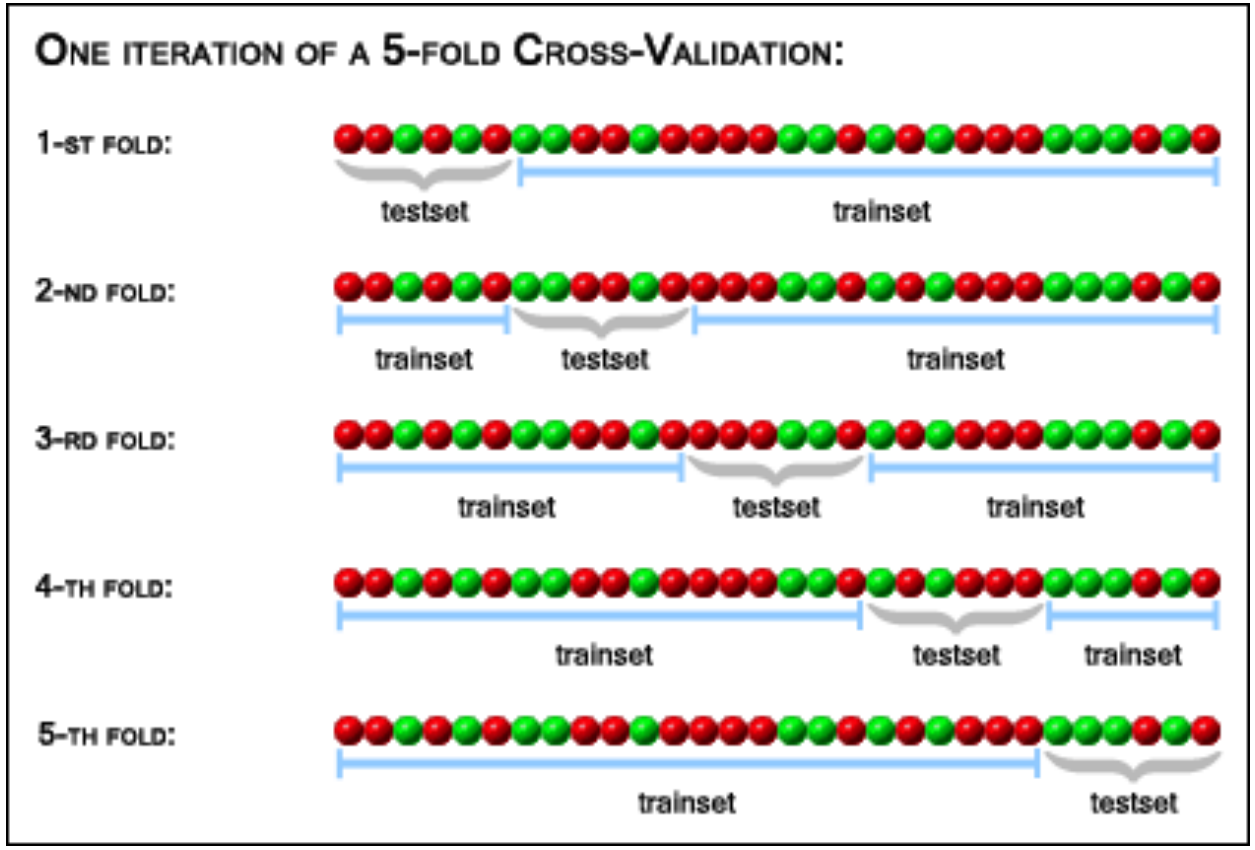
BASIC WORKFLOW

1. Clean the data / Normalize / Explore the data (visualize the data). This can be a cycle; visualize, notice weird things, clean up, visualize again, clean some more. This process helps you get familiar with the dataset.
2. Train / Test / Validation split
3. Train and finetune a model. This means selecting an apropriate model, fitting it, making predictions and finetune your hyperparameters.

Cross-validation

CROSS-VALIDATION

A process of repeated splitting, training and testing. Often used for **model selection** & **parameter tuning**.



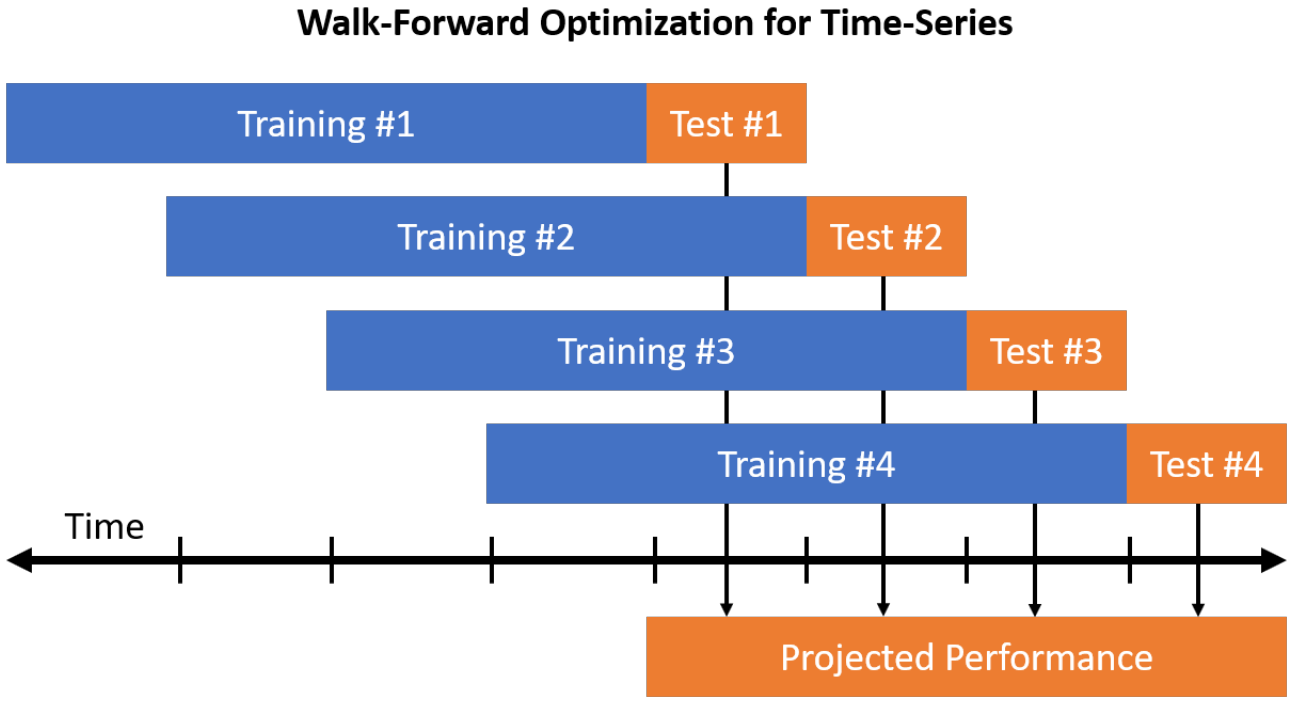
Advantages

- Generalize to all datapoints in the dataset
- Use the available data more efficiently

Check the variance in performance over folds!

CROSS-VALIDATION STRATEGIES

- **Stratified k-fold**: ensure each fold is representative in terms of class frequencies
- **Group k-fold:** ensure that the same group is not represented in both training and testing sets
- **Leave one-out:** each split a single data point is used as the test set → computationally expensive
- **TimeSeriesSplit:** Walk-forward. (What would happen with a k-fold? What is the problem with that?)



CROSS-VALIDATION

Why is this usefull?

To what problems can cross-validation be a solution?

Grid-search & pipelines

GRID SEARCH

- Tuning the parameters of a model can improve the model generalization performance.
- **Grid Search**: try all possible combinations of the parameters of interest
- Can be computationally expensive…

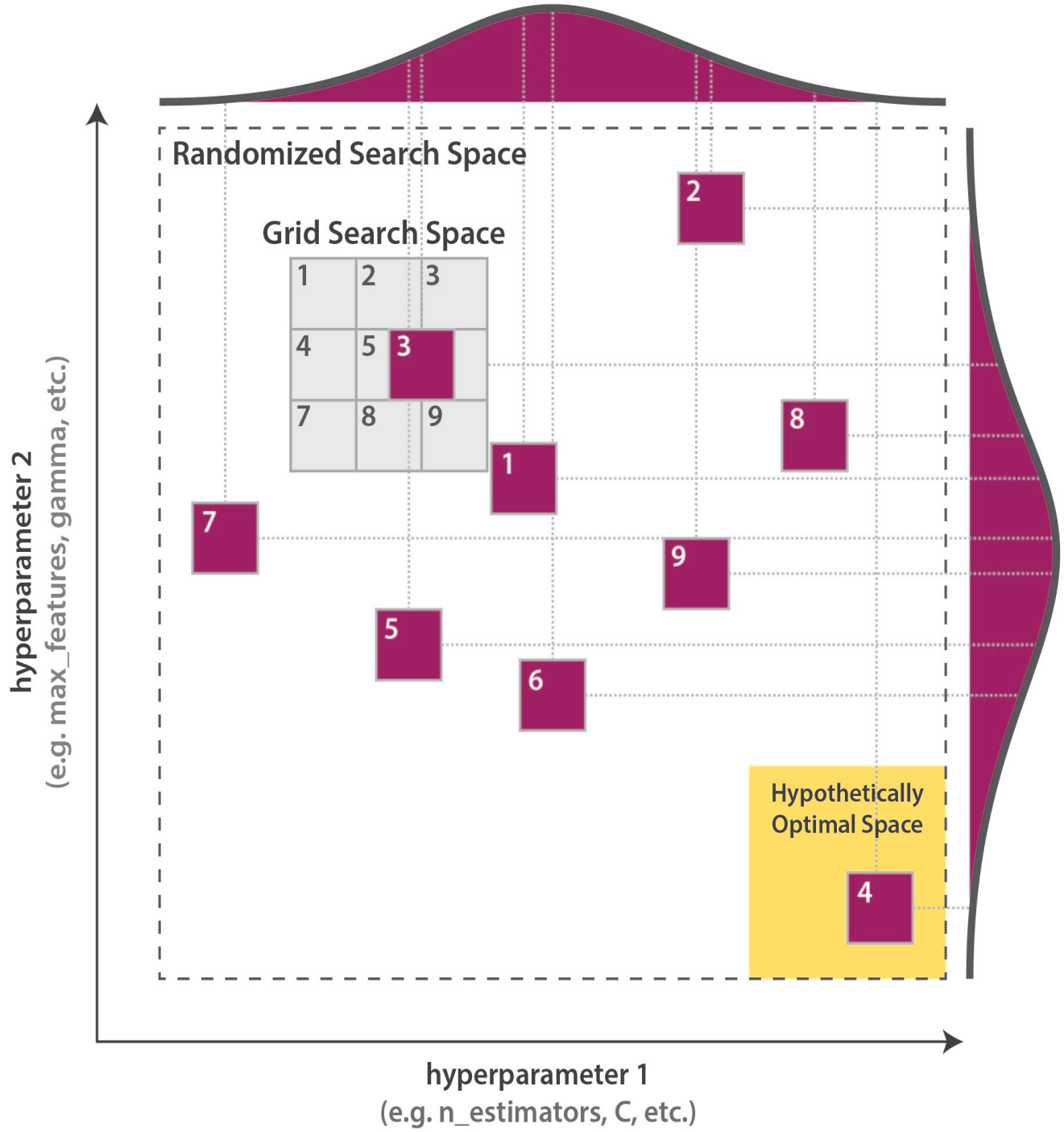
GRID SEARCH

This is the finetuning of your model. Most problems with the model start way before this stage.

E.g. the datascientist that made huge errors in feature selection will never recover from that with a grid search.

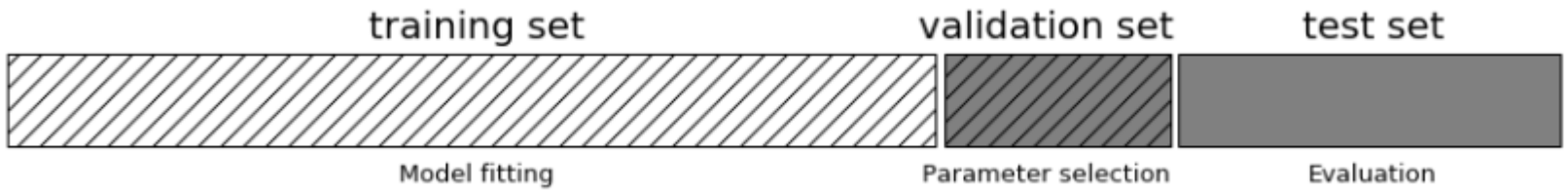
RANDOMIZED GRID-SEARCH

- Sample parameter values a certain number of times from some distribution which you prespecify in advance.

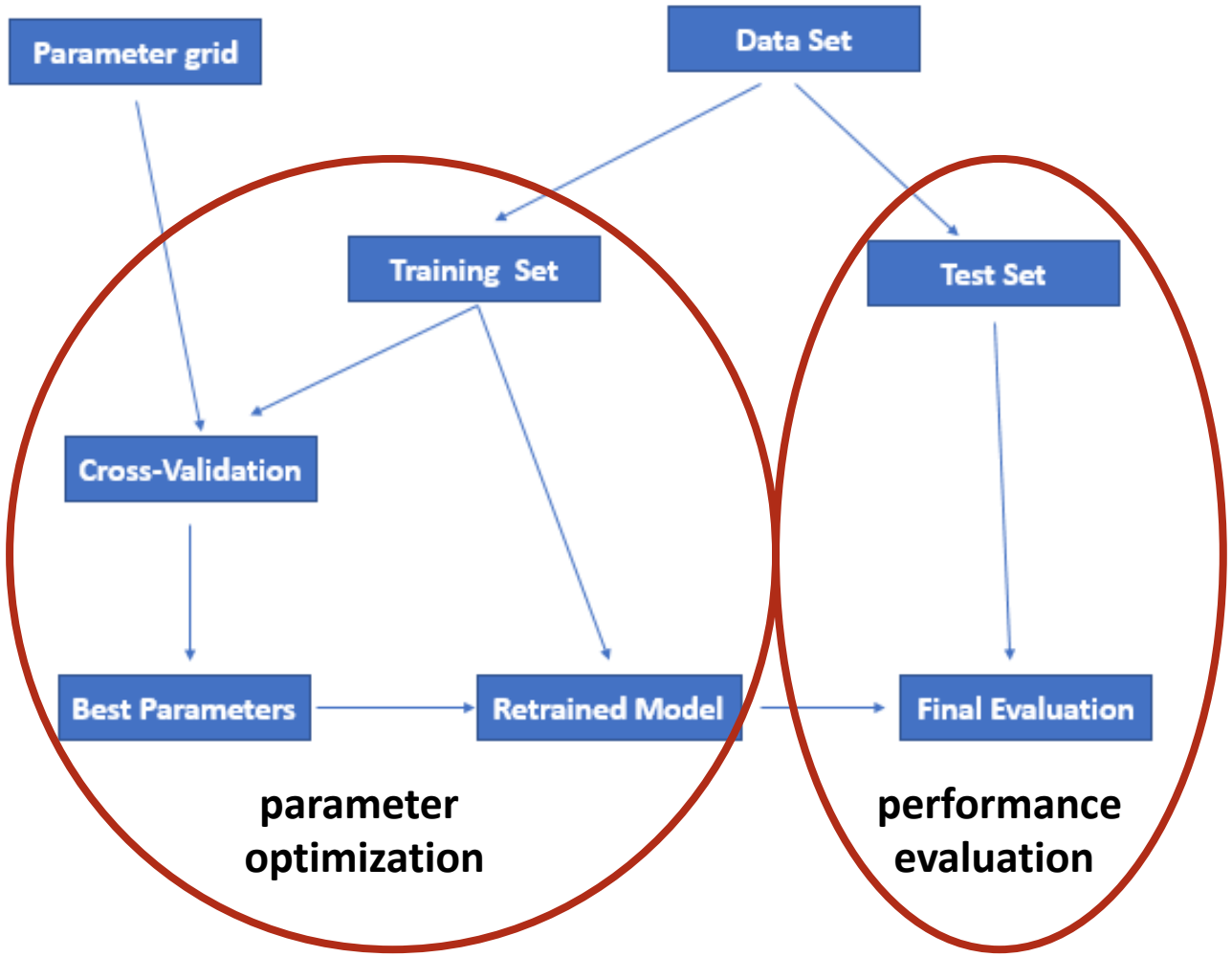


INFORMATION LEAKAGE

- **Only evaluate performance on ‘untouched’ data…**
- Information leakage can occur with:
  - Parameter selection
  - Data normalization

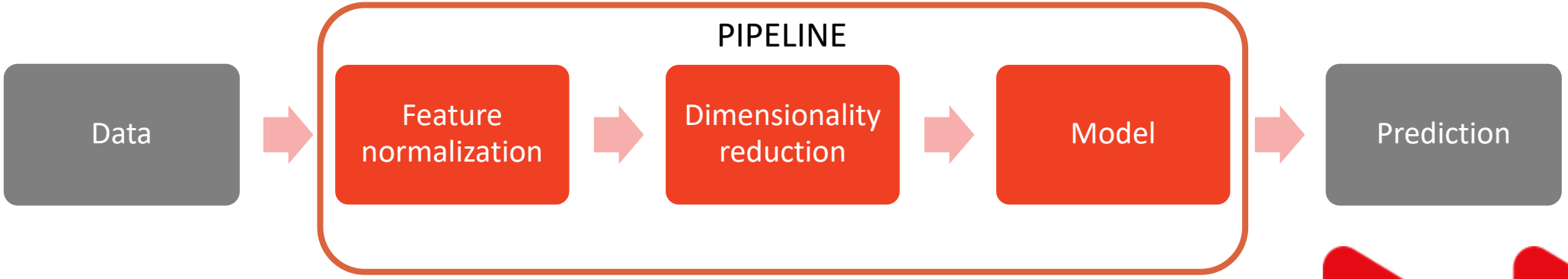


GRID SEARCH & CROSS-VALIDATION



PIPELINE

- **A pipeline combines any number of transformers plus one estimator**
- Machine learning often requires chaining together multiple processing (transformation) steps
- A pipeline itself is indistinguishable from an estimator (with the familiar .fit(), .predict() interface)
- Pipelines can be used easily in combination with GridSearchCV (useful to avoid information leakage)



LAB:

GRID-SEARCH & PIPELINES

ANCHORMEN
data activators

THANK YOU FOR YOUR ATTENTION

Cross-validation, Grid-search & Pipelines

Pedro de Medinalaan 11,
1086 XK Amsterdam

020 - 773 1972

www.anchormen.nl