ML Principles & Practical Issues

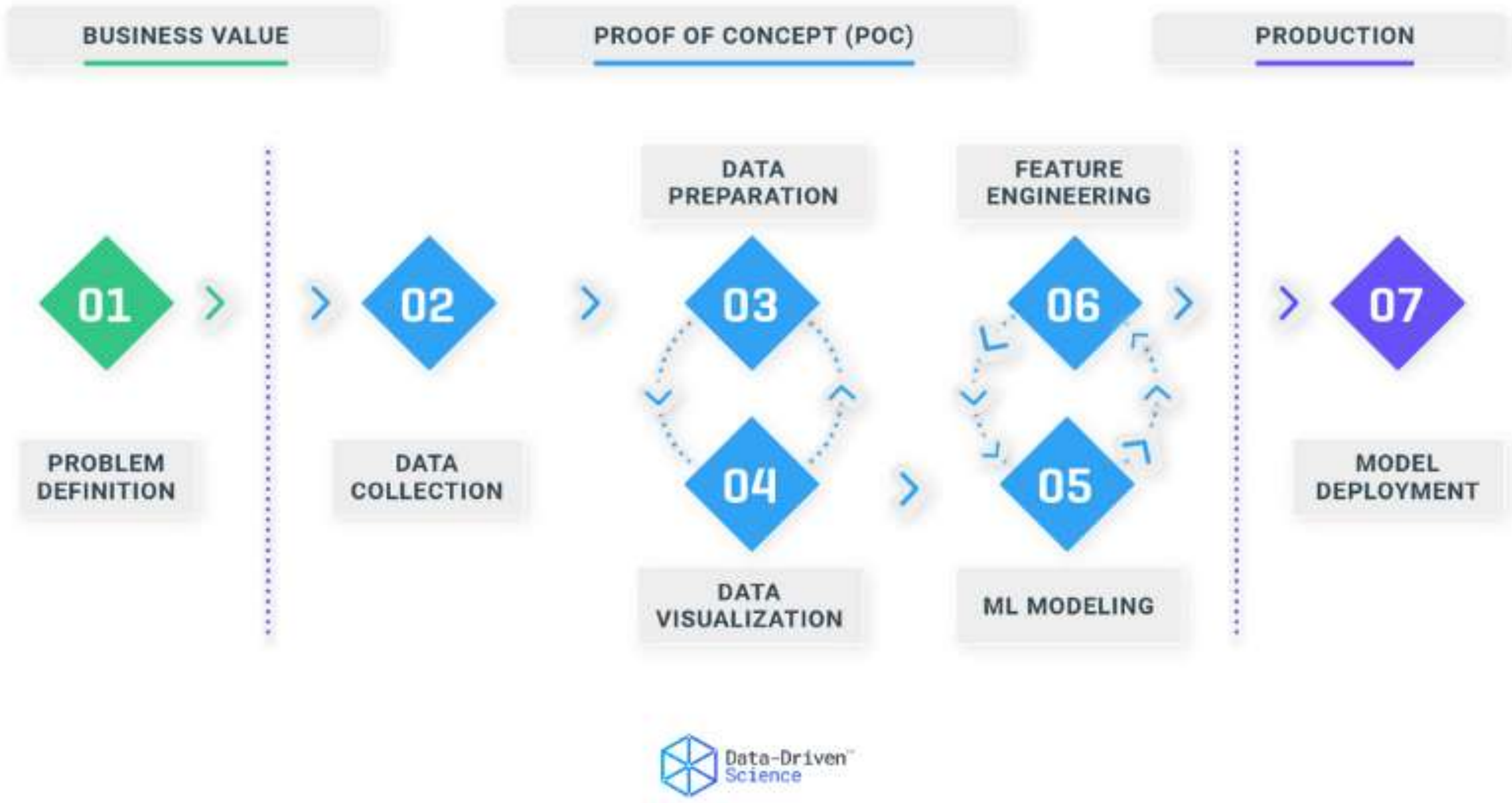
The characteristics of good ML problems

- Clear use case

- Relevant data

- Decision making

The ML Pipeline



https://medium.com/@datadrivenscience/7-stages-of-machine-learning-a-framework-33d39065e2c9

A Hypothetical case study

- A lab scientist wants to build an automated system that will allow her cat in and out of her office window and disallow dogs from entering through it

ML Problem formulation

- ML Problem Statement: Classify cats vs dogs correctly

Supervised Learning

$$y = f(x)$$

Predicted label
- cat
- dog

Input features from
- Images of cats and dogs

Machine learning function that maps input to output

Build an ML classifier

- Collect data

What kind? | How much? | From which source? | Time and Expense/ Quality?

- Pre-process the image data

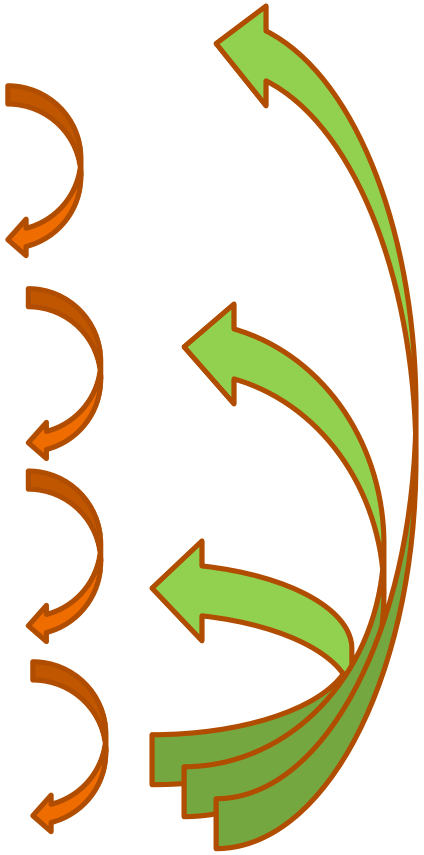
Resizing | Denoising

- Feature generation and data modelling

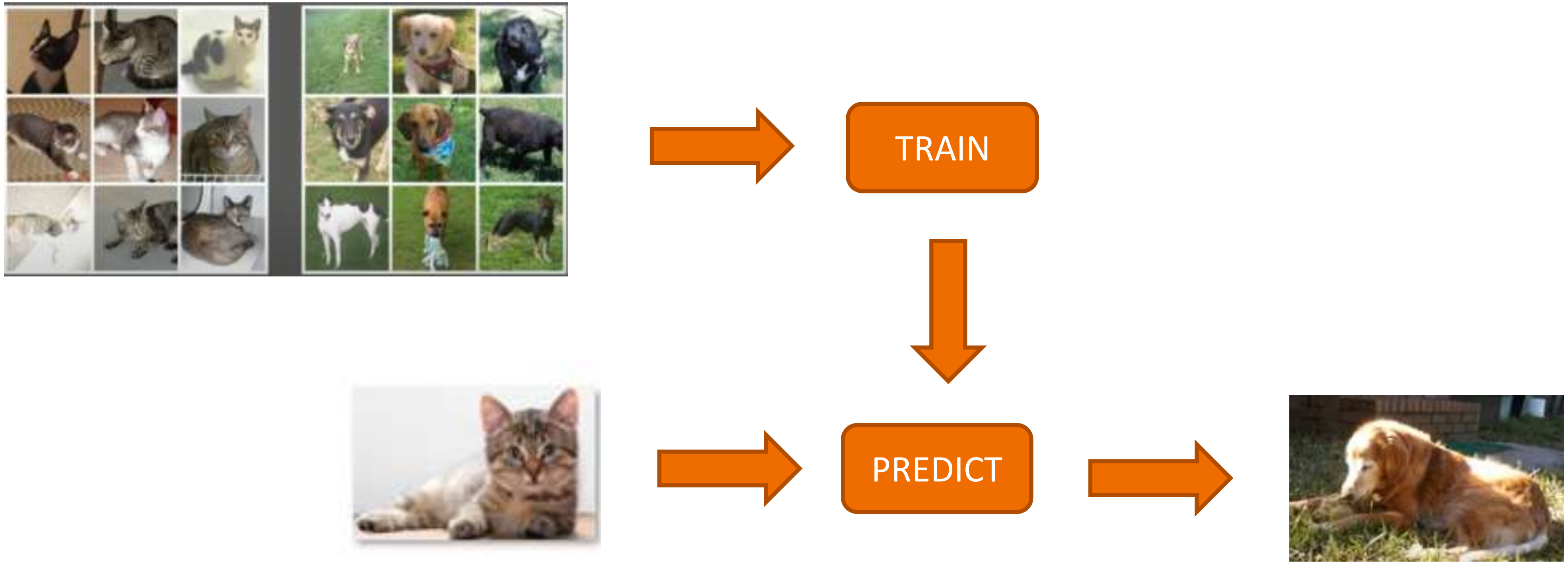
Handcrafted/ Statistical/ Deep learnt features | Which classifier?

- Evaluate the model

Which accuracy metrics?



Practical issues



New Sample

Predict
using trained
model

Output

What could be the reasons?

Data

- Noisy / low quality data?
- Insufficient data volume?
- Poor data pre-processing?

Features

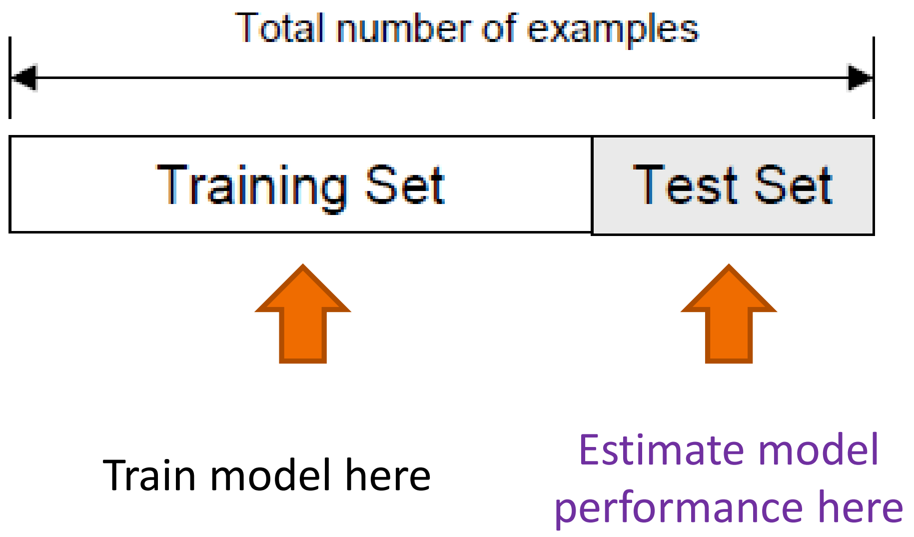
- **Scaling required?**
- **Feature selection/ extraction required?**

Model Training and Testing

- Insufficient training?
- Excessive training?
- Tuning of model parameters required?
- Algorithm needs to be changed?
- Testing method?

Holdout test set: The naïve approach

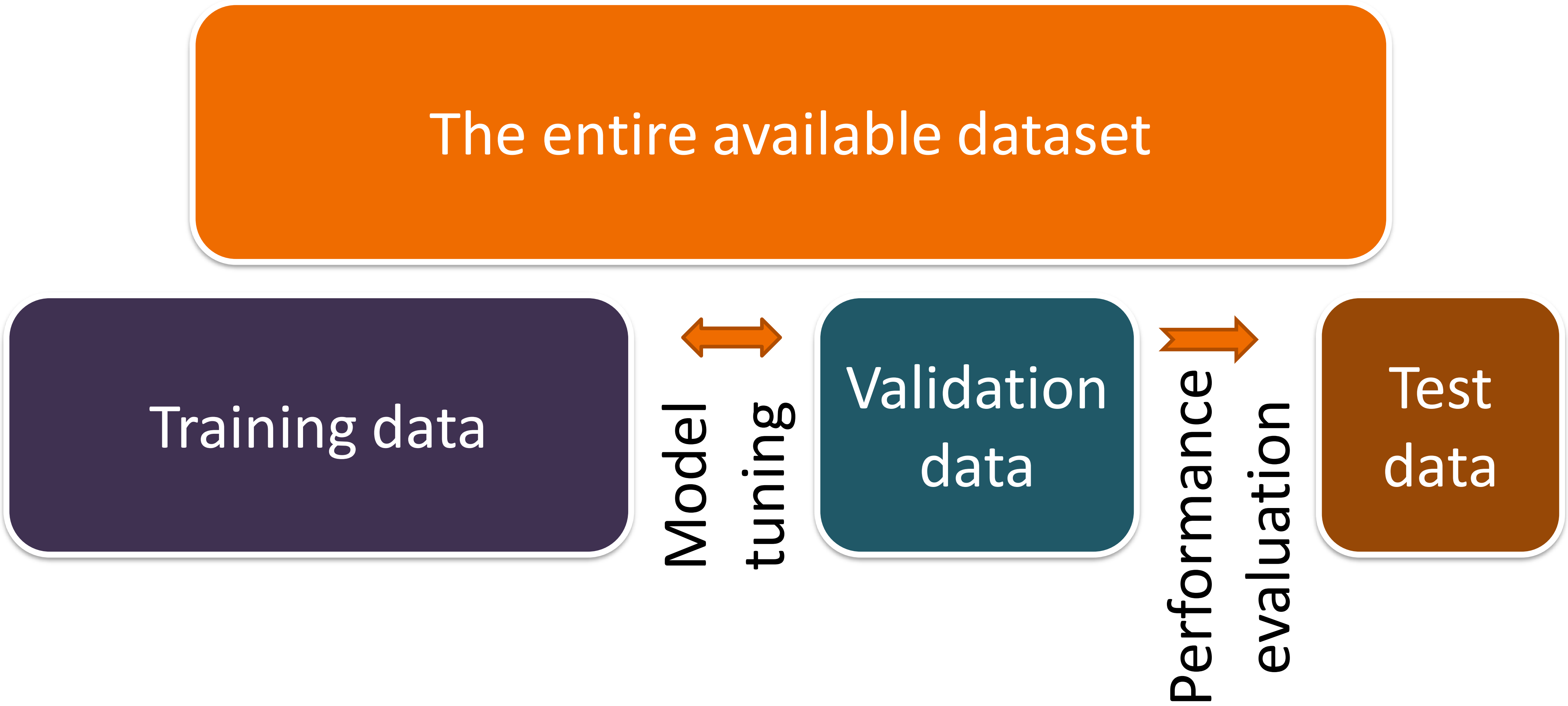
- Randomly split the entire dataset into:
  - **Training set:** A dataset used for training the model
  - **Test set (a.k.a validation set):** Data only used for testing the model



The three-way split

- **Training set**
  – A set of examples used for learning
- **Validation set**
  – A set of examples used to tune the parameters of a classifier
- **Test set**
  – A set of examples used only to assess the performance of a fully-trained classifier.

The three-way split



How to perform the split?

- How many examples in each data set?
  - **Training:** Typically 60-80% of data
  - **Test set:** Typically 20-30% of your data set
  - **Validation set:** Around 20% of data
- Examples
  - **3 way:** Training: 60%, Val: 20%, Test: 20%
  - **2 ways:** Training 70%, Test: 30%

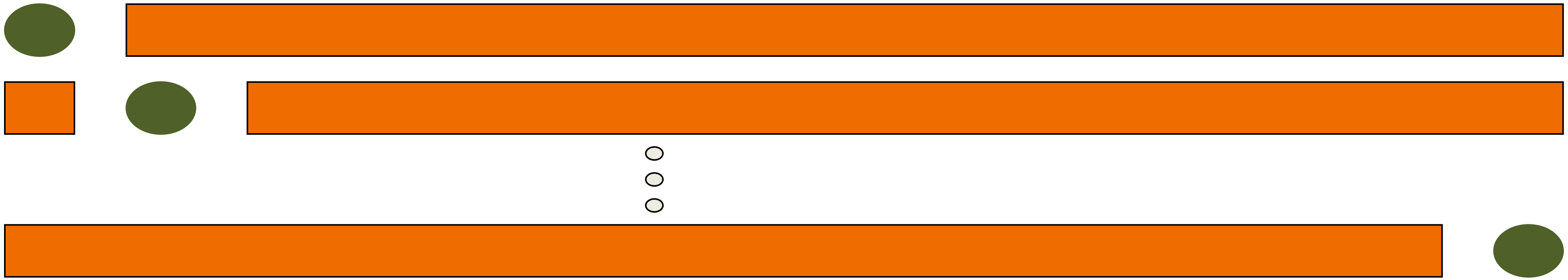
Holdout summary

- **Positive**
  - – Intuitive; Usually easy to perform; Considered the ideal method for evaluation
- **Drawbacks:**
  - – In small datasets, you do not have the luxury of setting aside a portion of your data
  - – The performance will be misleading if we had an unfortunate split

Common Splitting Strategies

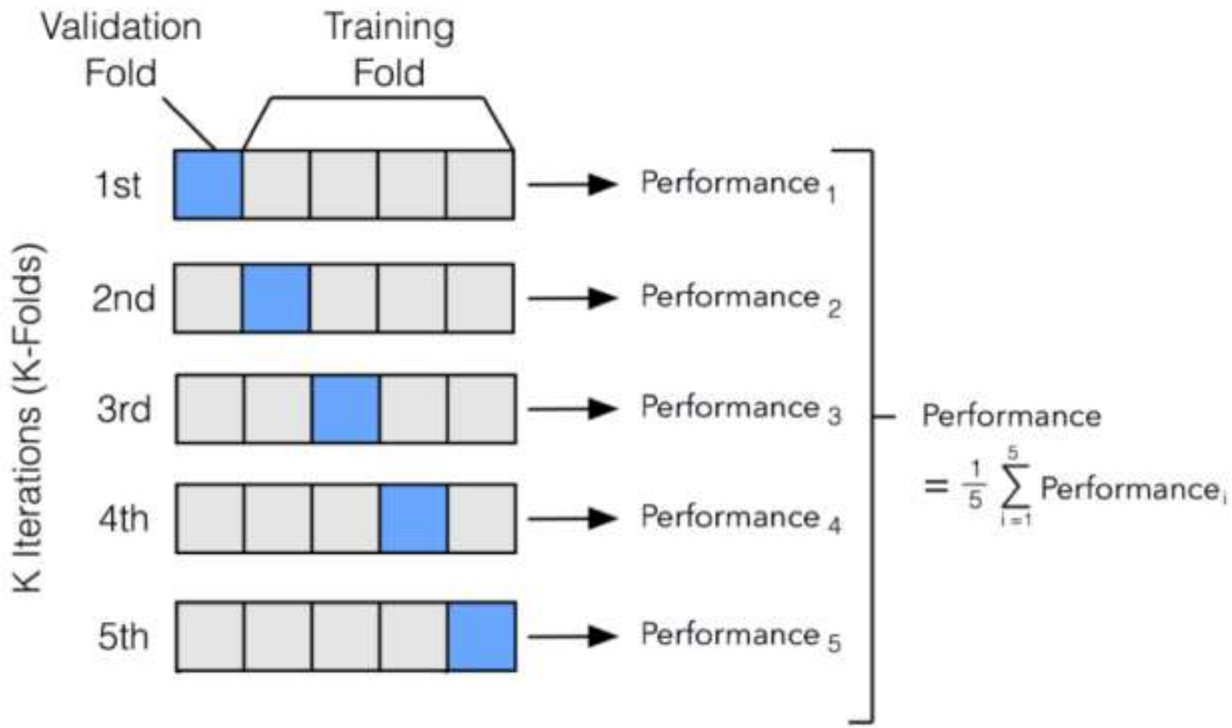
- k-fold cross-validation

- Leave-one-out (n-fold cross validation)



Estimating model performance

K-fold cross validation



- The dataset is split into K partitions of equal size
- k-1 folds are used to train a model, and the holdout kth fold is used as the validation set
- This process is repeated and each of the folds is given an opportunity to be used as the holdout test set. A total of k models are fit and evaluated, and the performance of the model is calculated as the mean of these runs.

- Small datasets

- Computationally more expensive

Cross - Validation

- How do you summarize the performance? $E = \frac{1}{K}\sum_{i=1}^{K} E_i$
  - **Average:** Usually average of performance between experiments
- How many folds are needed?
  - Common choice: 5-fold or 10-fold cross-validation (Some nice numbers)
  - Large datasets → even 3-Fold cross-validation will do
  - Smaller datasets → bigger K. Why?
  - Leave-One-Out approach (K=n). K is equal to the number of examples n. Used for very small datasets

Thanks!!

Questions?