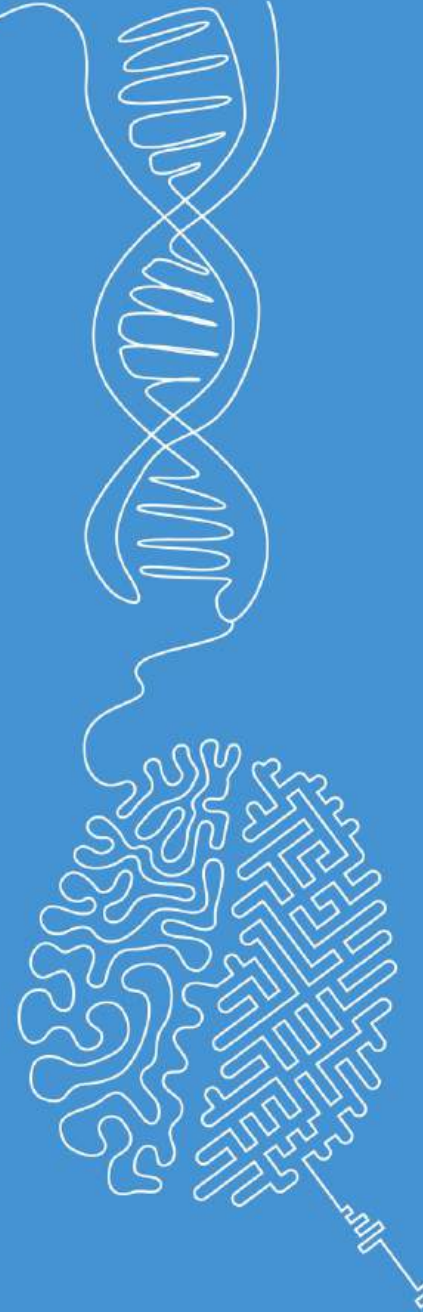


Data splitting

Machine Learning

Norman Juchler



Miscellaneous

Stuff to discuss before the lecture

Project work: Tips for selecting the problem

- Choose a “relevant” problem
- Choose a “solvable” problem
- Ideas to find public datasets:
 - [Kaggle](#)
 - [Google dataset search](#)
 - [Open data portal](#) (Bundesamt für Statistik)
 - [List of datasets for ML](#) (Wikipedia)
 - ...
- Practical tips:
 - Don't work with too many features (<30)
 - Make sure to have enough data per feature. Rule of thumb: 7-10 samples per feature...
 - Avoid categorical variables that have an excessive number of categories.
 - For features: One-hot encoding increases problem dimensionality
 - For target variables: Multiclass classification can become difficult

Quiz from last week

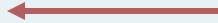
Why is feature engineering important?

Multiple answers are possible.

- ☐ a. It optimizes the number of features.
- ☒ b. It improves the model's ability to capture relevant patterns in the data.
- ☒ c. It helps to avoid the effects of the curse of dimensionality.
- ☒ d. It decreases model overfitting.
- ☐ e. It removes missing values from the dataset.

Quiz from last week

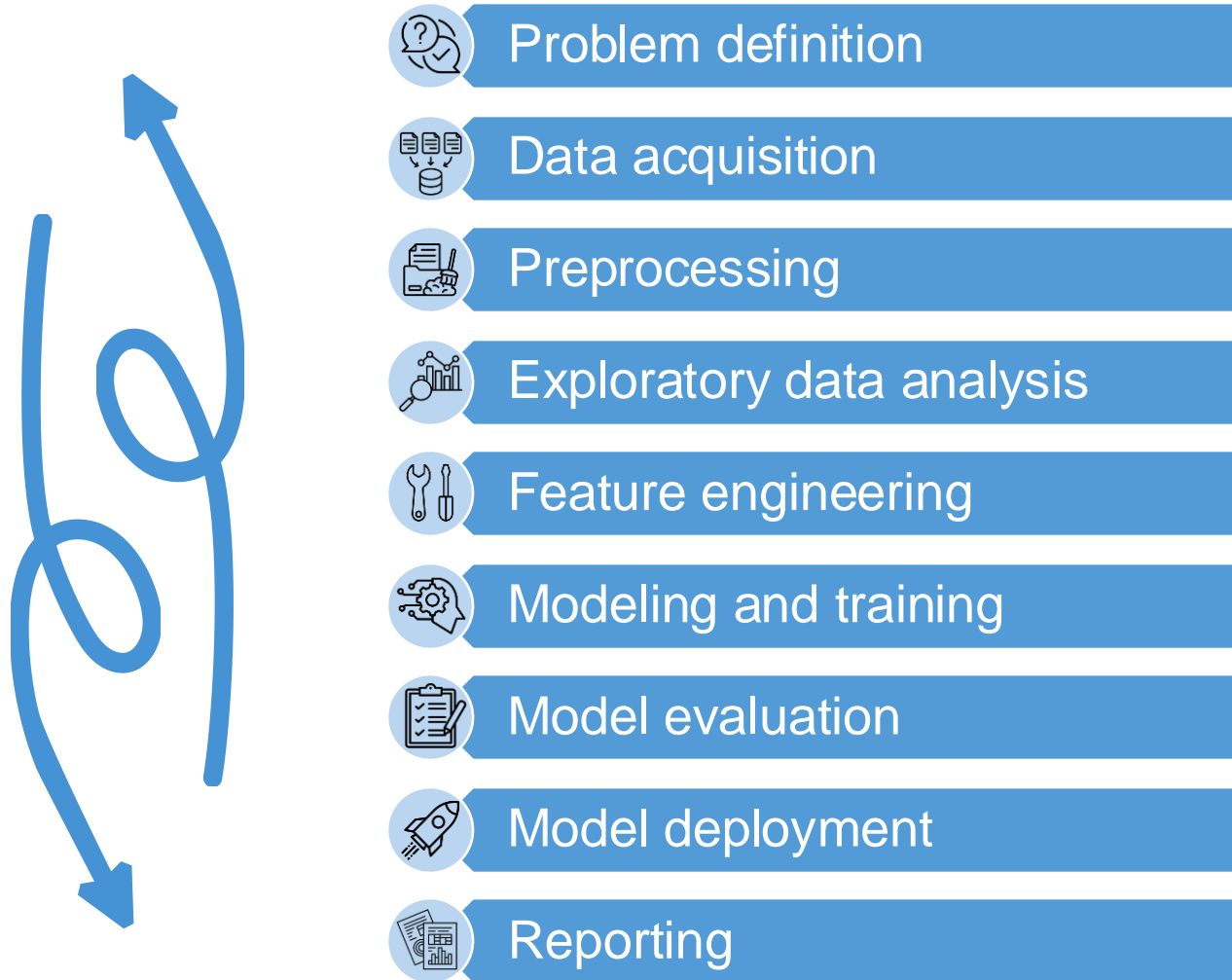
Which of the following is a common preprocessing task?

- ☒ a. Handling missing data
- ☐ b. Model training
- ☐ c. Feature engineering 
- ☒ d. Feature scaling
- ☐ e. Hyperparameter tuning
- ☐ f. Cross-validation
- ☒ g. Outlier removal
- ☒ h. Encoding categorical variables
- ☒ i. Converting data into a suitable format
- ☒ j. Data normalization or standardization
- ☐ k. Visualizing data

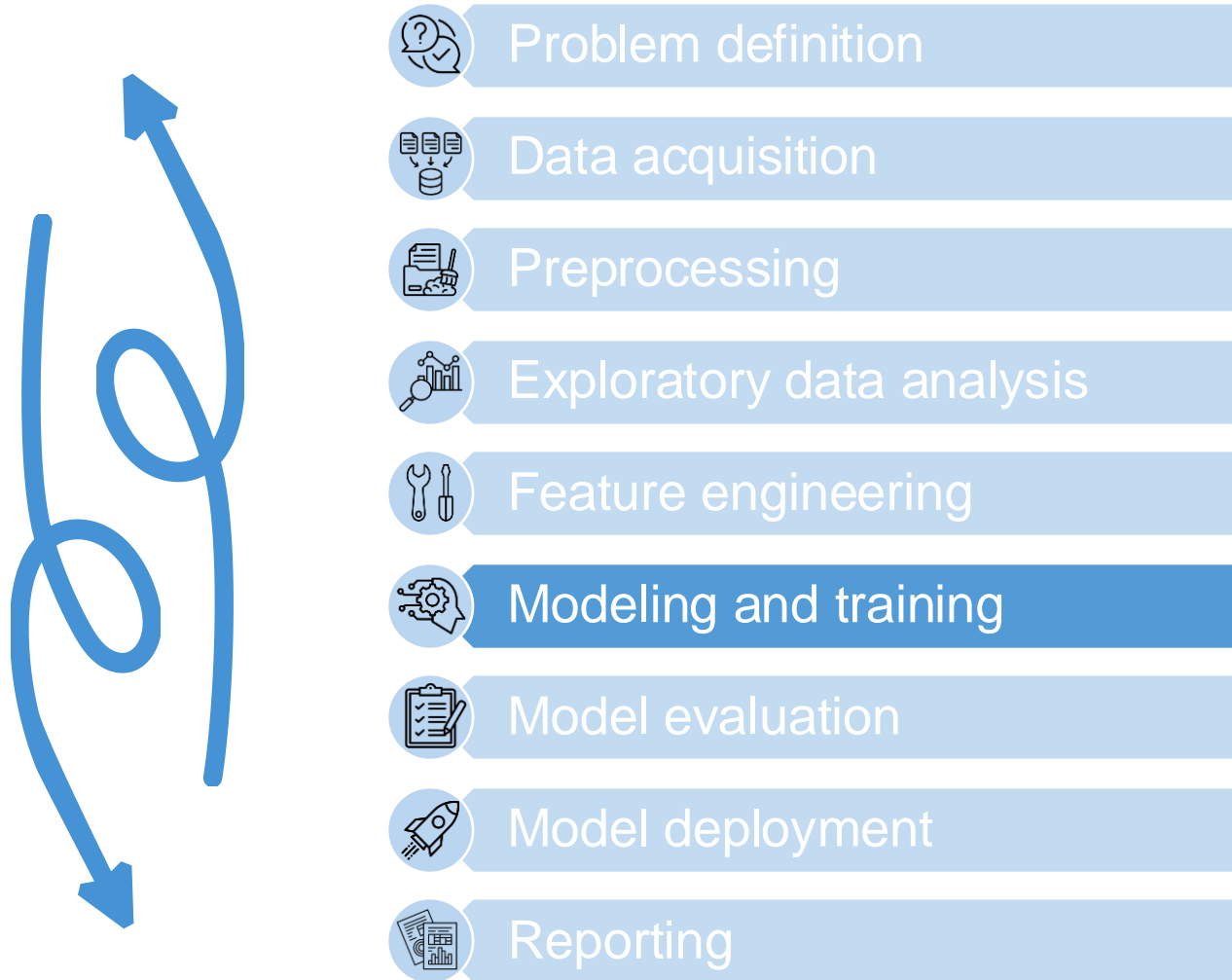
Data splitting

...in more depth

The data science workflow

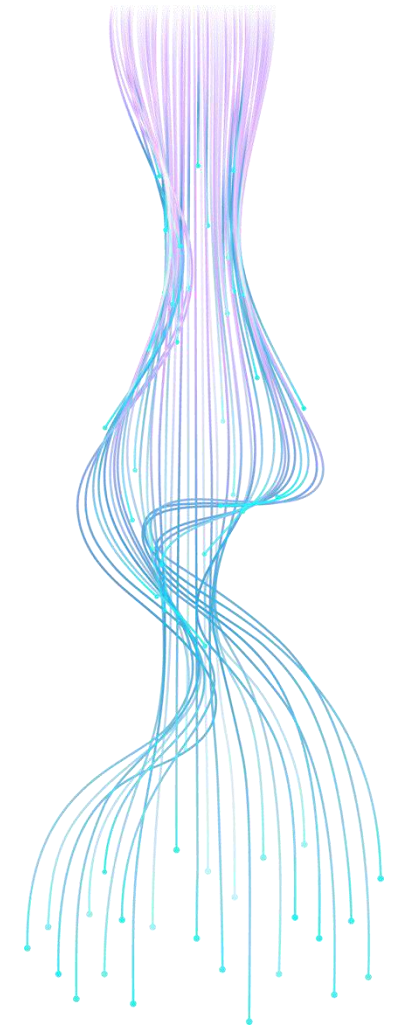


The data science workflow



Data splitting

- **To evaluate model performance:** Splitting data helps assess how well the model performs on unseen data, preventing misleading performance metrics.
- **To estimate the generalization error:** By measuring the difference between the model's prediction performance on the training and test data, we can assess how well the model will perform on new data.
- **To recognize overfitting:** By separating training and test data, we can detect if a model has learned irrelevant patterns or memorizes the training data.
- **To tune hyperparameters:** Validation sets enable hyperparameter tuning without biasing the model to the test data.
- **To avoid data leakage:** Proper splitting prevents data leakage, where information from the test set leaks into the training process, leading to overly optimistic performance estimates. Don't fool yourself or others. 😊
- **To ensure fairness:** Splitting ensures a fair evaluation of different models, by testing them on the same test data that wasn't used for training.



Method 1: No splitting – **VERY BAD PRACTICE!!!**

Dataset

- **Training:** Train a model using the entire data set
- **Testing:** Evaluate the model using the same data set
 - Use the model to compute predictions
 - Compute performance scores based on predictions and known true values
- **This is bad practice!**
 - Overfitting to the data set is likely to go unnoticed.
 - The performance metrics will be biased towards the dataset.
 - Unseen data is very likely to lead to poorer results → poor real-world performance

Method 2: Train-test split



- Divide the dataset into **training** and **test sets**
 - Common ratios: 4:1 (80% training vs 20% testing) or 3:1 (75% vs. 25%)
 - It is often recommended to shuffle the data beforehand.*
 - Goal: The resulting datasets preserve the properties of the original dataset
- **Training:** Train the model using the training set
- **Testing:** Evaluate the model using the test set
 - Use the model to compute predictions, and compare those with the known true values
 - Make sure that the training and test sets are strictly separated

* Different shuffling strategies are possible. For classification problems, the target labels can be ignored (random shuffling) or shuffled such that the ratio of labels in the training and test sets is maintained (stratified shuffling).

Method 3: Train-validation-test split



- Conceptually very similar to the train-test split
- Split the training set once more to obtain a training set and a validation set.
- **Training:** Train the model using the training set
- **Validation:** Use the validation dataset to find a good set of hyperparameters (=special *configuration* parameters that are not learned by the model)
- **Testing:** Evaluate the model using the test set

Method 4: Cross-validation (for evaluation)

	Dataset				
Split 1:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 1
Split 2:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 2
Split 3:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 3
Split 4:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 4

- Can be seen as a generalization of the train-test split
- K-fold cross-validation:
 - Shuffle the data randomly
 - Split the entire dataset into k equally sized partitions
 - Create k different splits, assign folds to training and test sets (see above pattern)
 - Train and evaluate a model for each split

Method 4: Cross-validation (for evaluation)

	Dataset				
Split 1:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 1
Split 2:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 2
Split 3:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 3
Split 4:	Fold 1	Fold 2	Fold 3	Fold 4	→ Model 4

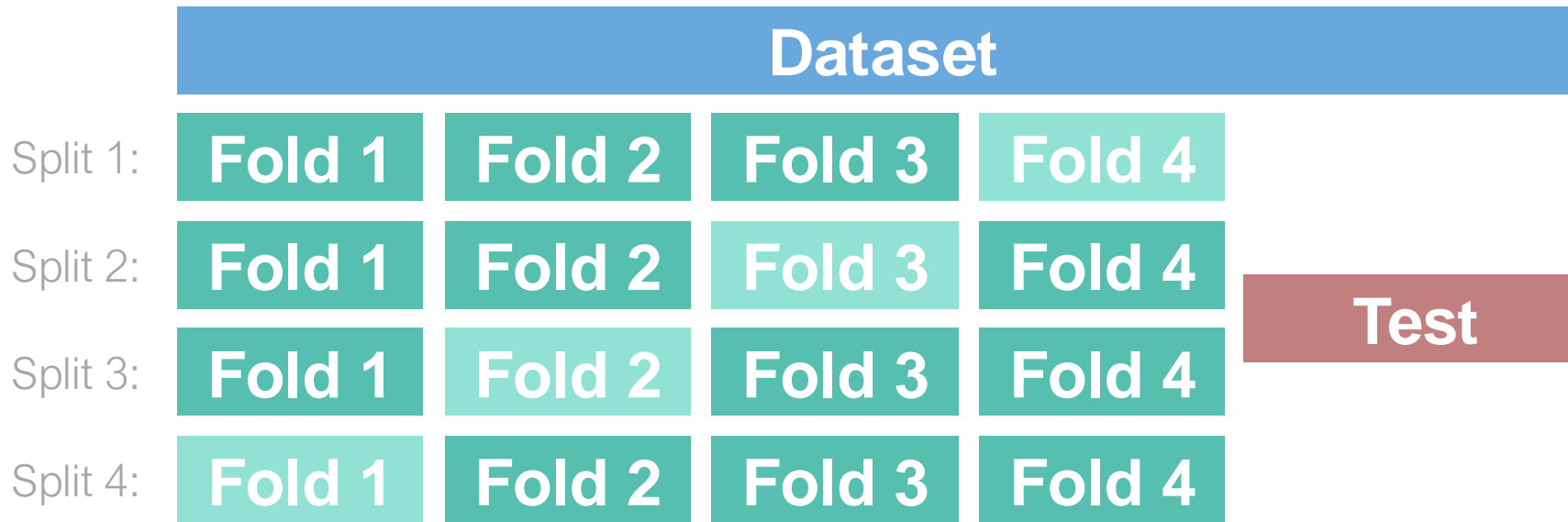
■ Advantages:

- Cross-validation allows a more robust assessment of the generalization error
- Model training is a random process → Provide performance metrics as mean and std of k models
- Efficient use of data: Use all data for both training and validation

■ Disadvantage:

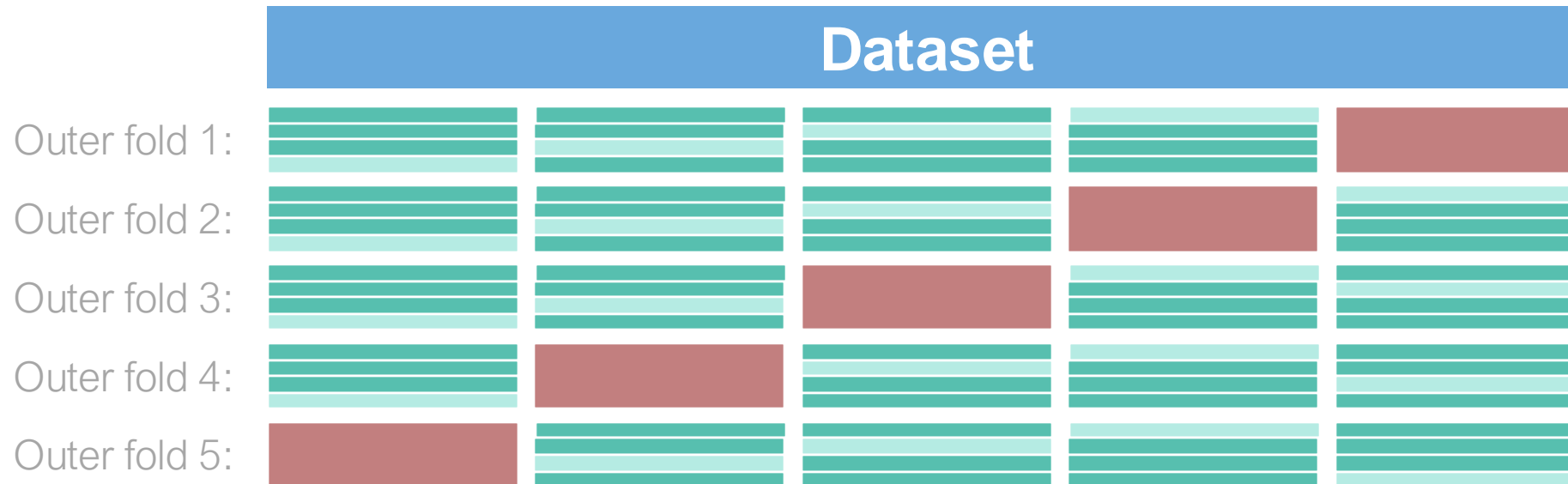
- Requires training and evaluation of k models

Method 4: Cross-validation (for hyperparameter tuning)



- Can be seen as a generalization of the train-validation-test split
- Motivation: Collect more evidence for finding a good set of hyperparameters
 - Often combined with “grid search” (comes later) or other search strategies
 - Choose a hyperparameter configuration that has the best average score on all splits

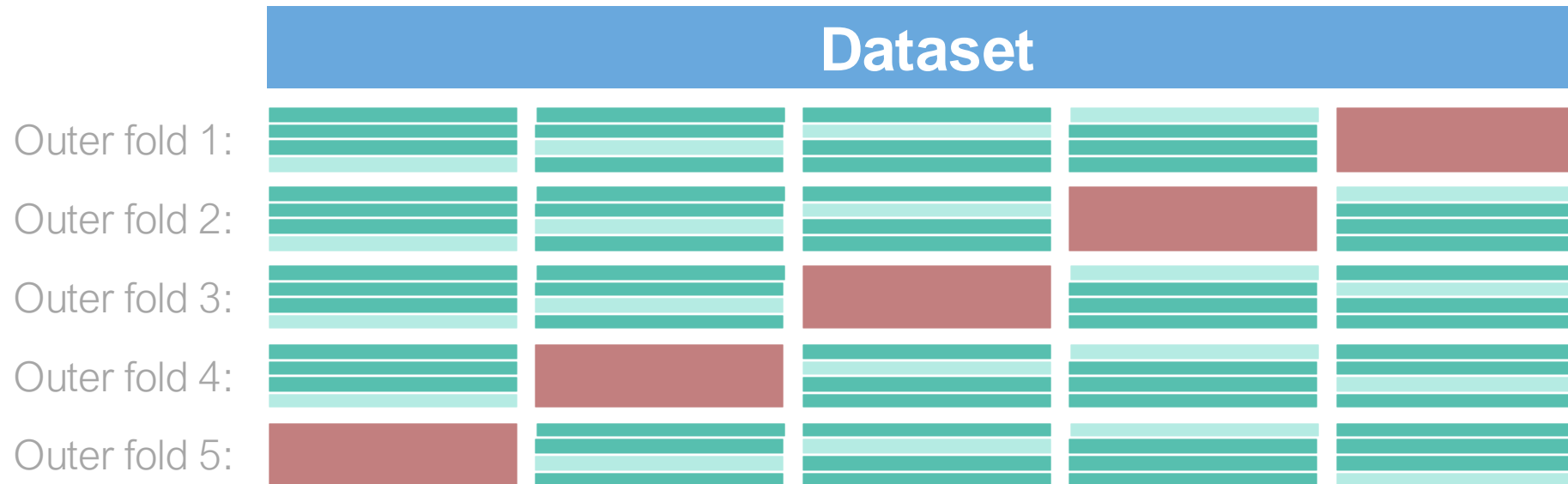
Method 5: Nested cross-validation



■ The recommended method...

- ...when hyperparameter tuning or model selection (see later) is involved
- ...when you want an **unbiased performance estimate**
- ...in high-stakes fields (e.g., medicine, finance) where accuracy and reliability are critical
- ...for small datasets where overfitting is a concern

Method 5: Nested cross-validation



- Not very practicable if model training is expensive!
 - Often the case for deep-learning
- Nested k-fold cross-validation:
 - Training: Involves $m \cdot (k-1)$ model trainings (m : number of inner folds, k : number of outer folds)
 - Testing: Involves k model evaluations

Summary

- Data splitting is an essential step in machine learning:
 - Detect overfitting / estimate the generalization error
 - Used in the context of hyperparameter tuning
 - Unbiased performance estimation
 - Model selection / comparison: Splitting helps compare different models and choose the best one based on their ability to generalize.