# Data Splitting

Divide a dataset into two or more subsets to train, validate, and evaluate a *Machine Learning* model.

- **Training Set**
  - Largest portion (70-80%)
  - **Goal**: Train the model (learn **patterns and relationships** in the data).

- **Validation/Development Set**
  - Small portion (10-15%)
  - **Goal**: Tune the model's hyperparameters and prevent **overfitting**.

- **Test Set**
  - Small portion (10-15%)
  - **Goal**:
    - Obtain an unbiased estimation of the trained model's performance.
    - The model **does not learn anything** from this set.

## Importance of Data Splitting

- **Prevent Overfitting**: By evaluating the model on data unseen during training (validation/test sets), we can check if the model has **learned generalizable patterns** (instead of **memorizing** the training data itself).

- **Model Selection and Hyperparameter Tuning**: The validation set allows us to compare different models and their configurations (hyperparameters) to choose the one that performs best on unseen data.

- **Assess Generalization**: The test set provides a final, unbiased estimate of the model's **ability to generalize to new data**.

### Data Ordering Bias

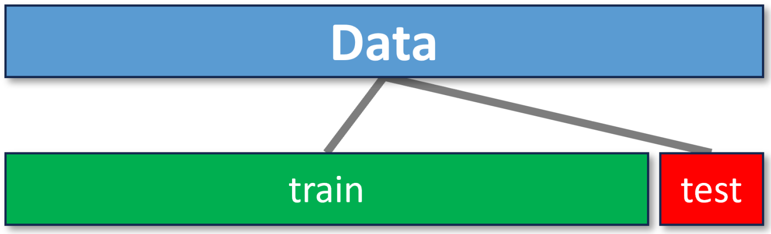
- Data might be ordered by class or any other feature.
- If data were collected in batches, they might have inherent similarities within each batch.
  - Data were ordered by collection time.

To prevent bias from data ordering, data must be randomly shuffled prior to any data splitting.

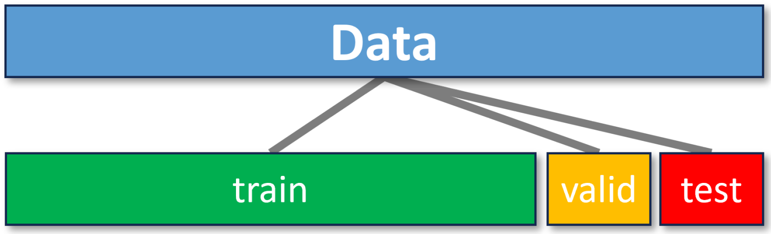
- **Shuffle first, then split**
  - Or use a splitting function that does the shuffling.
- Except for time series, where the temporal order is crucial.

## Data Splitting Techniques

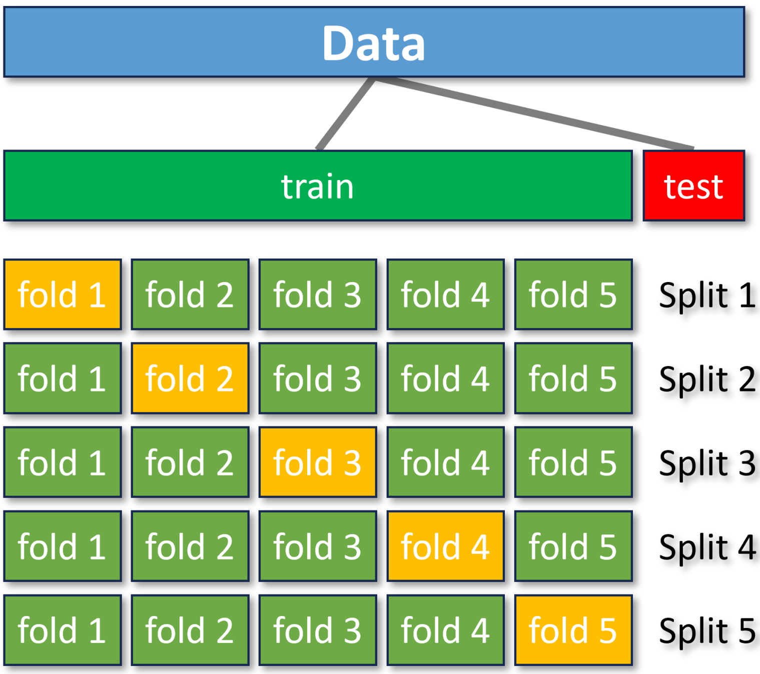
- **Train-Test Split**: The simplest (and possibly wrong) method. Does not allow hyperparameter tuning.




- **Train-Validation-Test Split**: The most common method. Divides the data into three sets for training, hyperparameter tuning, and final evaluation.




- **K-Fold Cross-Validation**: The dataset is first divided into train and test data sets. The train dataset is then further divided into $k$ equal-sized *folds*.
  - The model is trained and evaluated $k$ times, with each fold serving as the validation set once and the remaining folds used for training.
  - The performance is averaged across all k evaluations.
  - Provides a more robust estimate of performance, especially with smaller datasets.

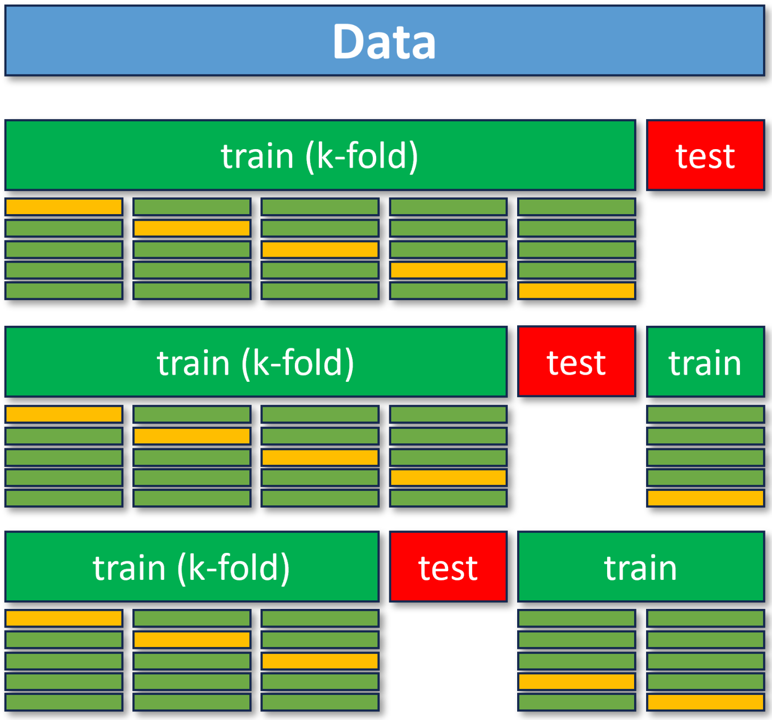



Final evaluation:

- Cross-validation is used to tune the model's hyperparameters (or select the best model)
- The model is trained on the entire training set

- The model is evaluated on the test set

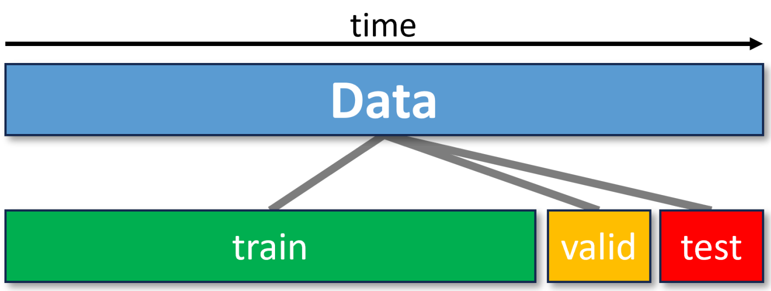
- **Nested Cross-Validation**:



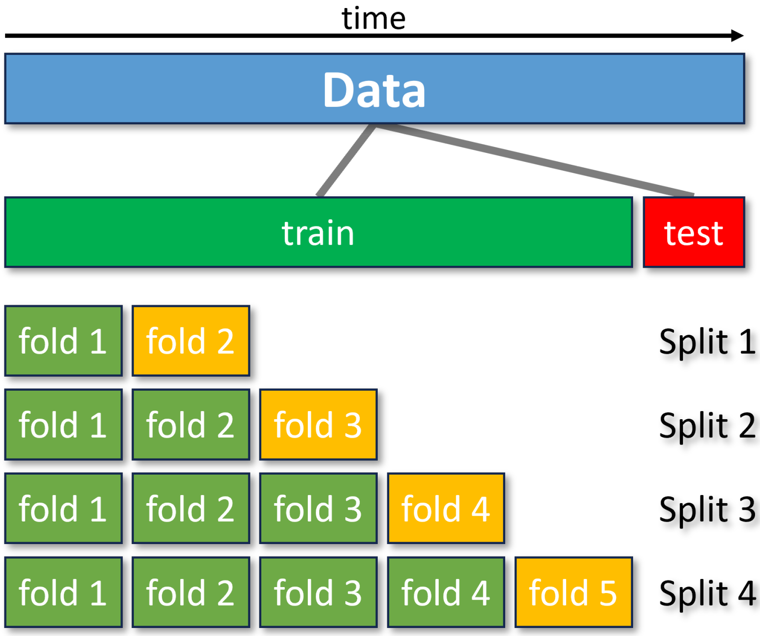

Some remarks on Nested Cross-Validation:

- It provides a less biased estimate of the true generalization performance.
  - Avoids overfitting
- Optimal hyperparameters found in the inner loop will likely be different for each outer fold.
  - You don't get a concrete model (with fixed hyperparameters)

- **Time Series Split**: Used for time-dependent data
  - The data is split chronologically (train → test)
  - Avoid *Lookahead Bias*




- **Time-Series Cross-Validation**
  - Avoid *Lookahead Bias*
  - Split data chronologically
  - For each validation fold, use just previous training data




Time-Series Cross-Validation with overlapping: