

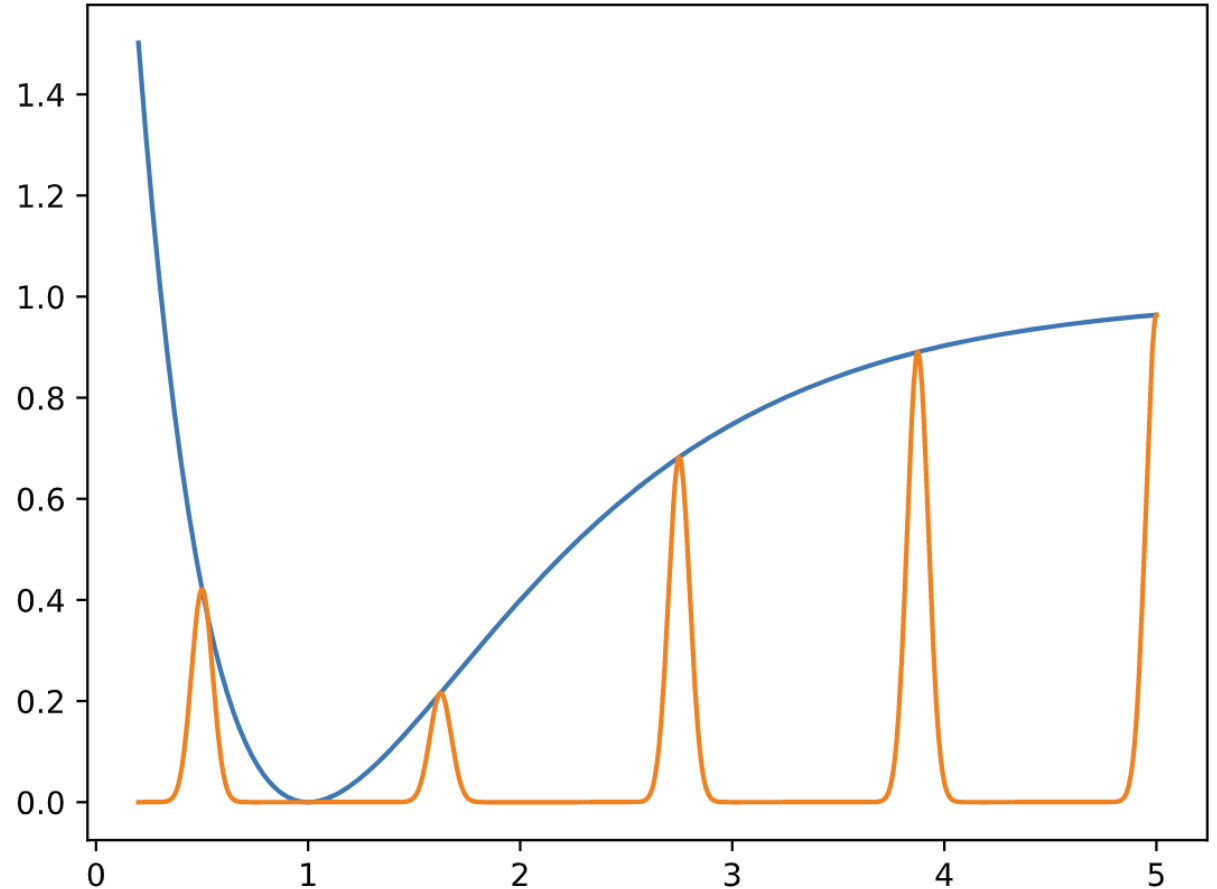
Cross-validation

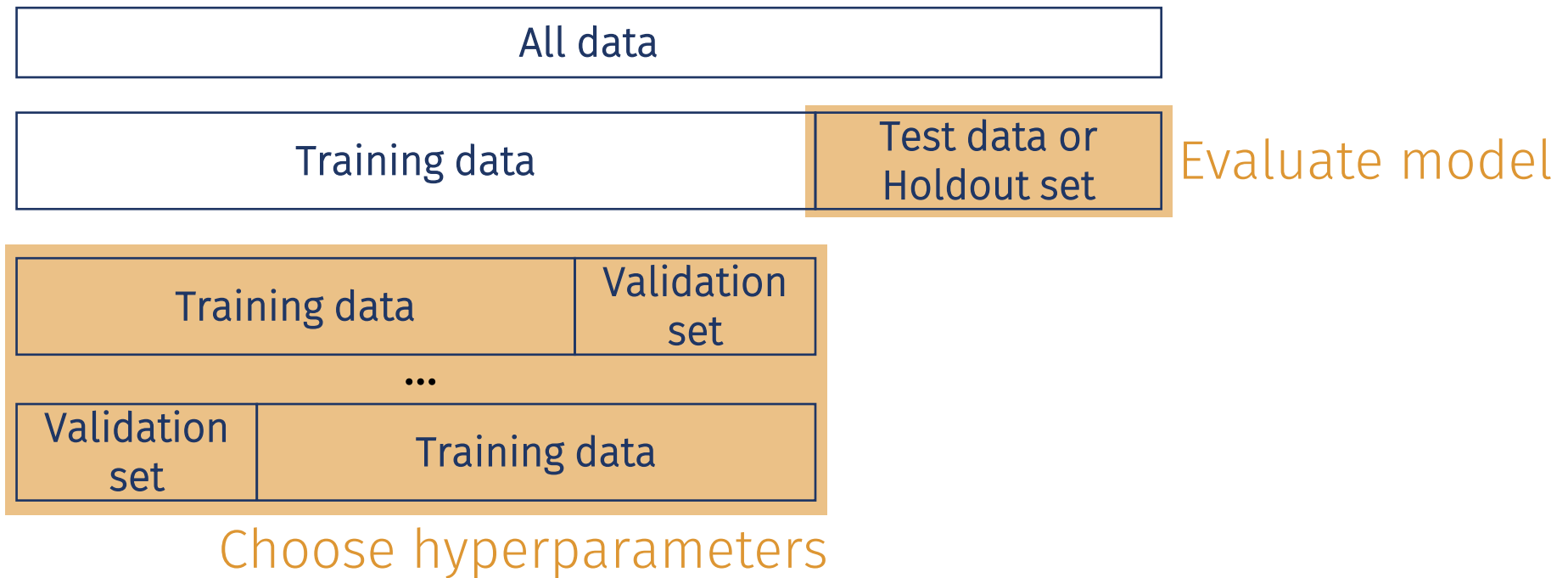
Challenge

- Need quality assessment
 - Cannot use training data
- Hyperparameters
 - Require unseen data

Solution

- Cross-validation: estimate the expected error if applied to unseen data (generalisation error)





Training points are independent and identically distributed (i.i.d.):

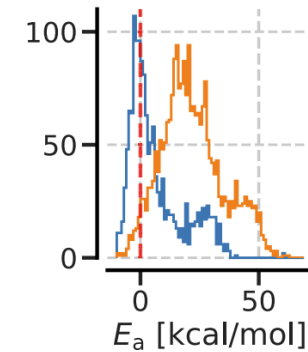
- k-fold most common: $k=5$ or 10
- Leave-p-out most common: $p=1$
- Shuffle split / Monte Carlo

Otherwise:

- Data specific
- Often: stratification first, then one of the above
- Generally: think about what constitutes unseen data and where information could leak into the model

Is it valid?

- Labels often not identically distributed
- Reference data not independent
 - Converged?
 - Fast enough?
 - Method worked?
 - Generated using assumptions



Method

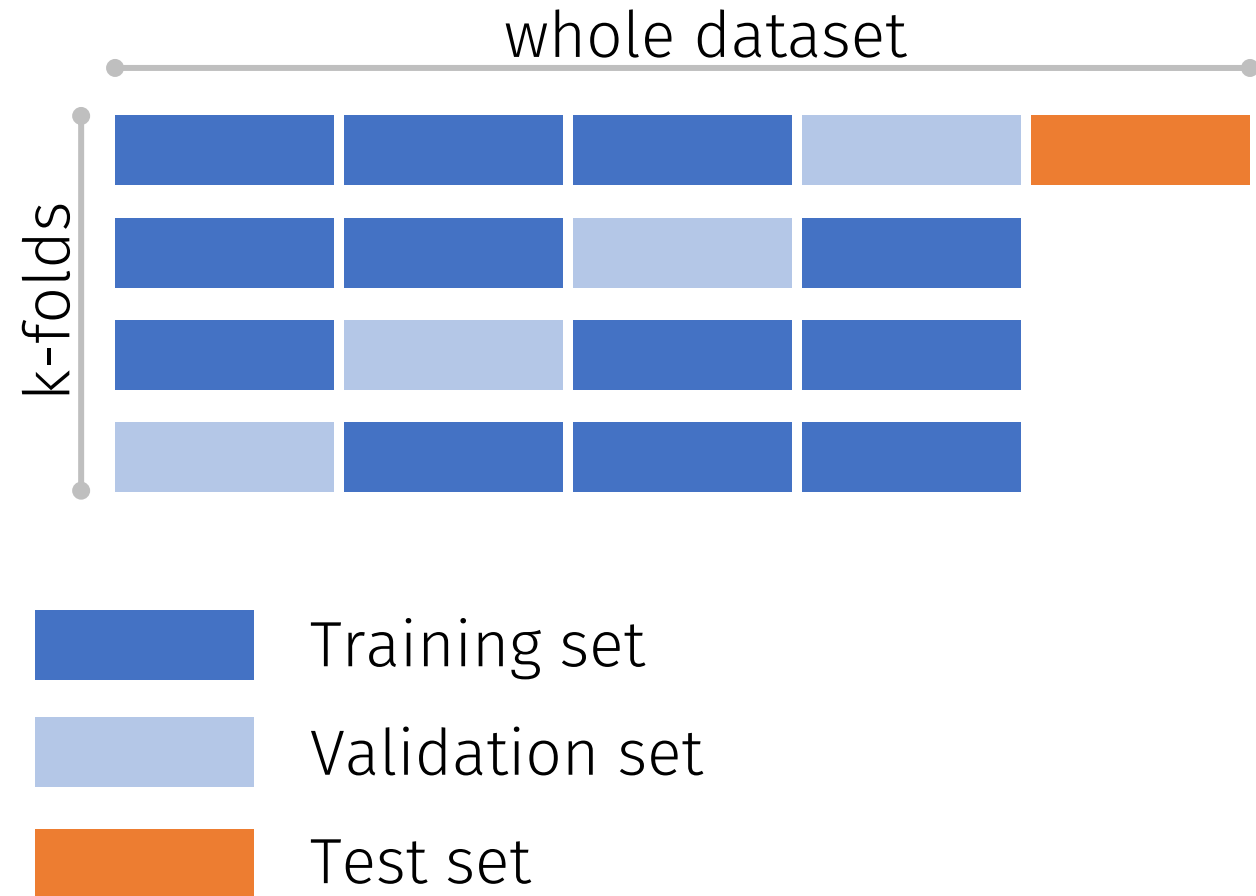
- Build n disjoint blocks
- Choose each block once as validation, train on rest
- Report metric on test set

Advantages

- Chooses all data points equally
- Converges quickly

Disadvantages

- Need to choose k
- Imbalanced blocks for most data sets
- Cannot reduce noise further



Method

- Use all but p entries for training, rest validation
- Do all combinations (expensive!)

Advantages

- Low noise, since exhaustive
- Chooses all data points equally

Disadvantages

- Quite expensive, quickly becomes infeasible

Method

- Use all but p entries for training, rest validation
- Do **some** combinations (choose p entries randomly)

Advantages

- Converges to leave- p -out
- Chooses all data points equally

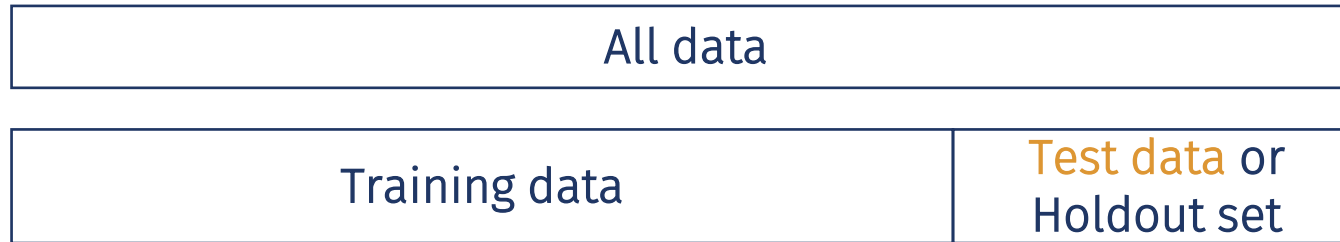
Disadvantages

- Needs quite few random selections to converge

- Training data could be imbalanced: Detecting vans in the city
- Subsets become imbalanced despite random selection: Sock drawer problem

Solution: Stratification

- Choose any subset / split s.t. it is closely representative of the full data set
 - Mean label / distribution of labels
 - Prevalence of groups
 - Prevalence of features
- Independent of method of cross-validation / actual model



Too small:

Bad estimate of generalisation error

Too large:

Loosing training data

Solution: fully nesting k-fold

- Average k-fold over all choices for test data

- Shuffle all data
 - Removes bias of ordering
- Stratify into 5 groups
 - Prevents overfitting
 - Helps if test set is small
- Repeat until converged:
 - For each of these groups:
 - One group as test set, rest as training
 - Test on all data points
 - Split train into stratified random sets
 - Estimate hyperparameters
 - Check whether hyperparameters and performance is converged

Summary Cross-validation

- Key target is reliable estimation of generalization error
- May require stratification to avoid bias
- Different methods differ in cost, simplicity, and convergence
- Relevant representatives: Leave-p-out, k-fold, shuffle split
- Common split ratio 80%/20% (no hard justification except experience)