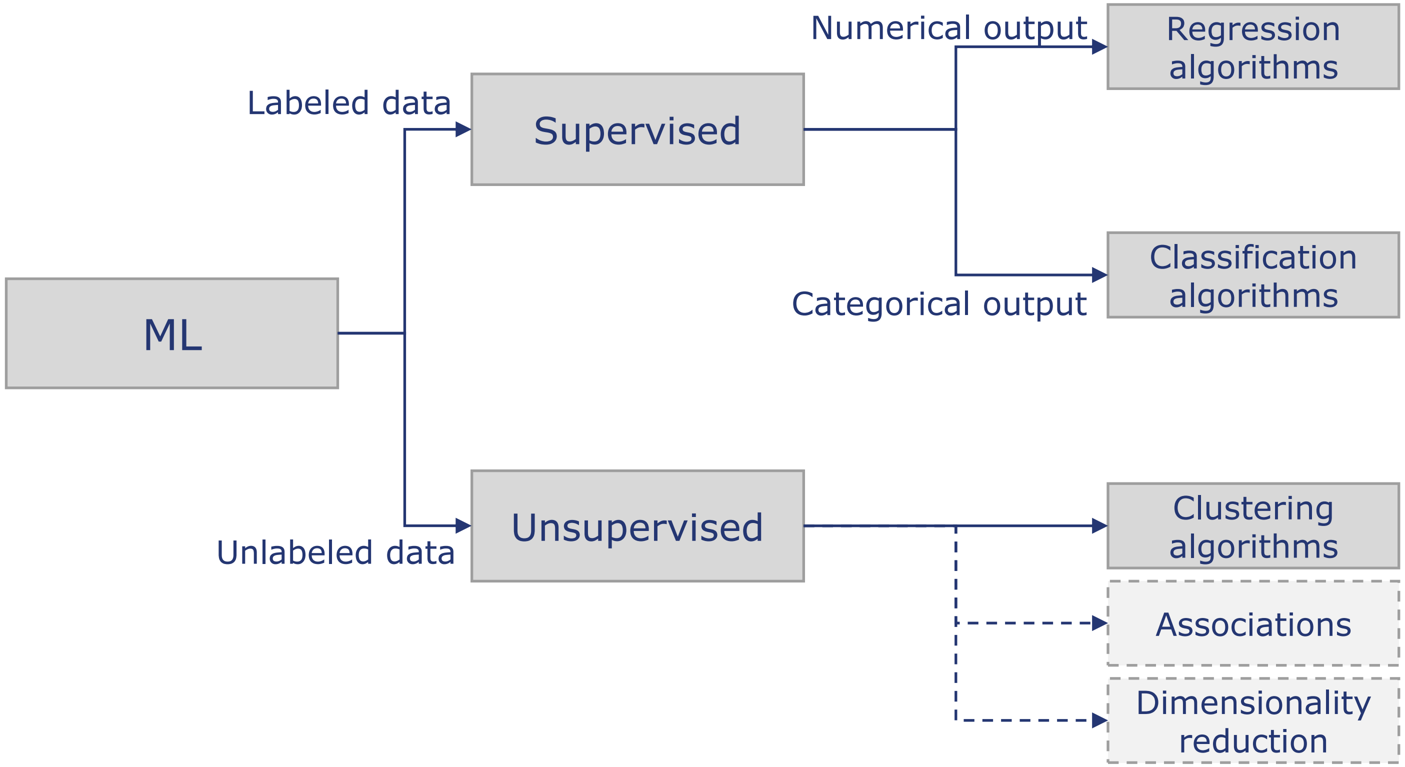
FH KREMS
UNIVERSITY OF APPLIED SCIENCES / AUSTRIA

Machine Learning, Artificial Intelligence, and Big Data Analytics (IL, 4th Semester)

Lecture 2

Recap of previous lecture (1/2)

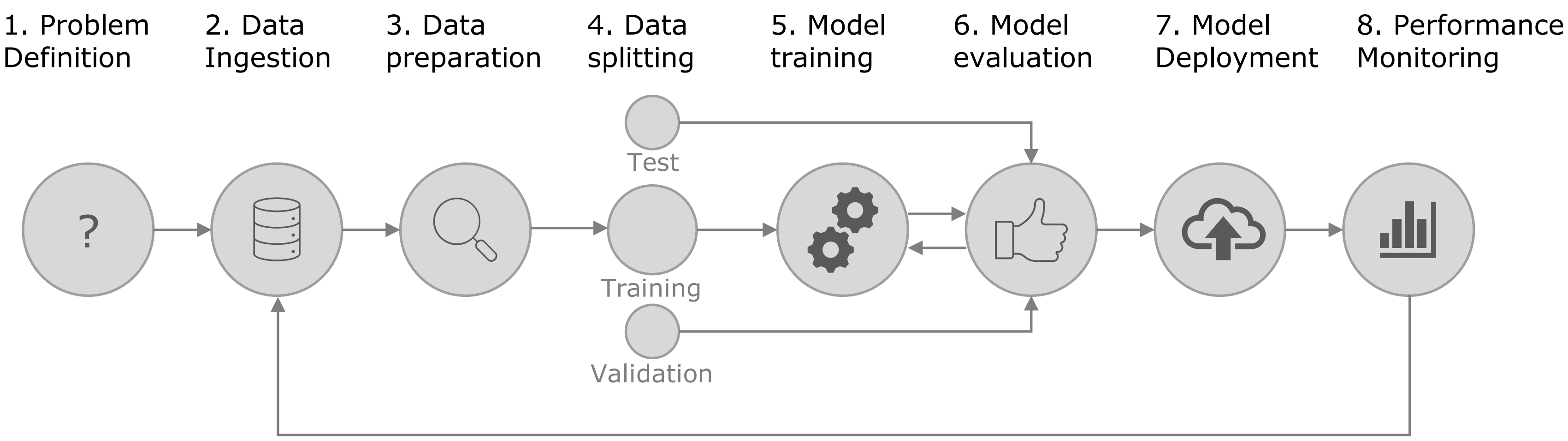


- Linear regression
    - Simple
    - Multiple
    - Multivariate
- Non-linear regression
- Decision Tree
- Random Forest

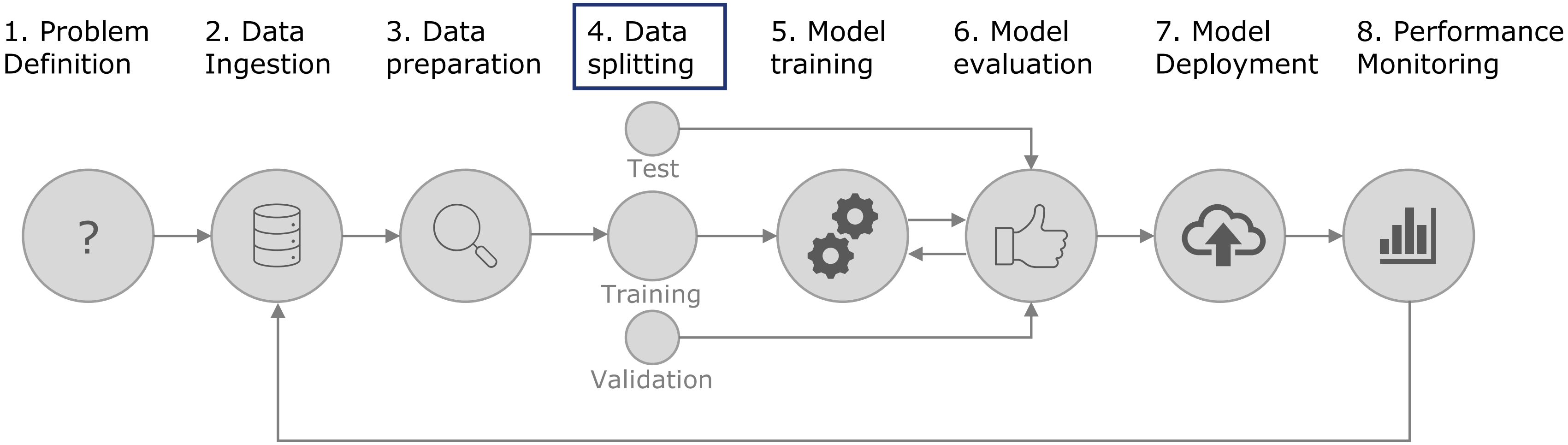
- Naïve Bayes
- Logistic regression
- Support Vector Machines
- K-nearest neighbors
- Decision Tree
- Random Forest
- …

- K-means
- Hierarchical clustering
- Density-Based clustering
- Model-based clustering
- …

Recap of previous lecture (2/2)



Focus of this lecture



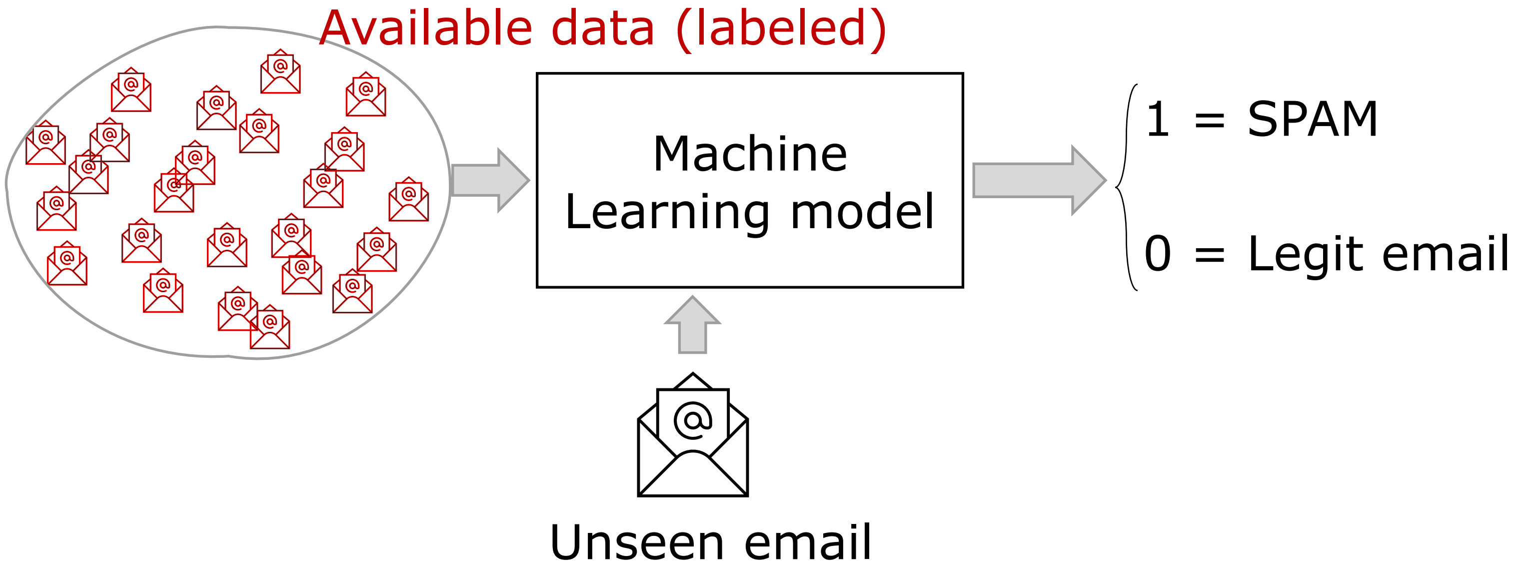
Topics of today

- Data Splitting
- Overfitting/underfitting
- The Bias-Variance tradeoff
- Cross-validation

Data splitting

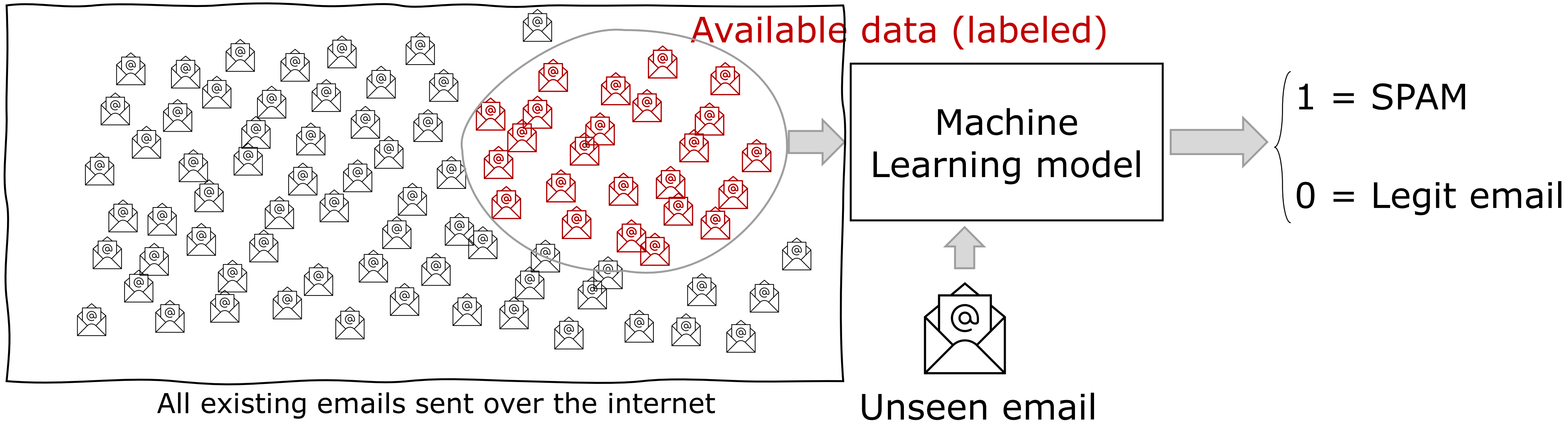
Why is it necessary?

Example:
SPAM detection



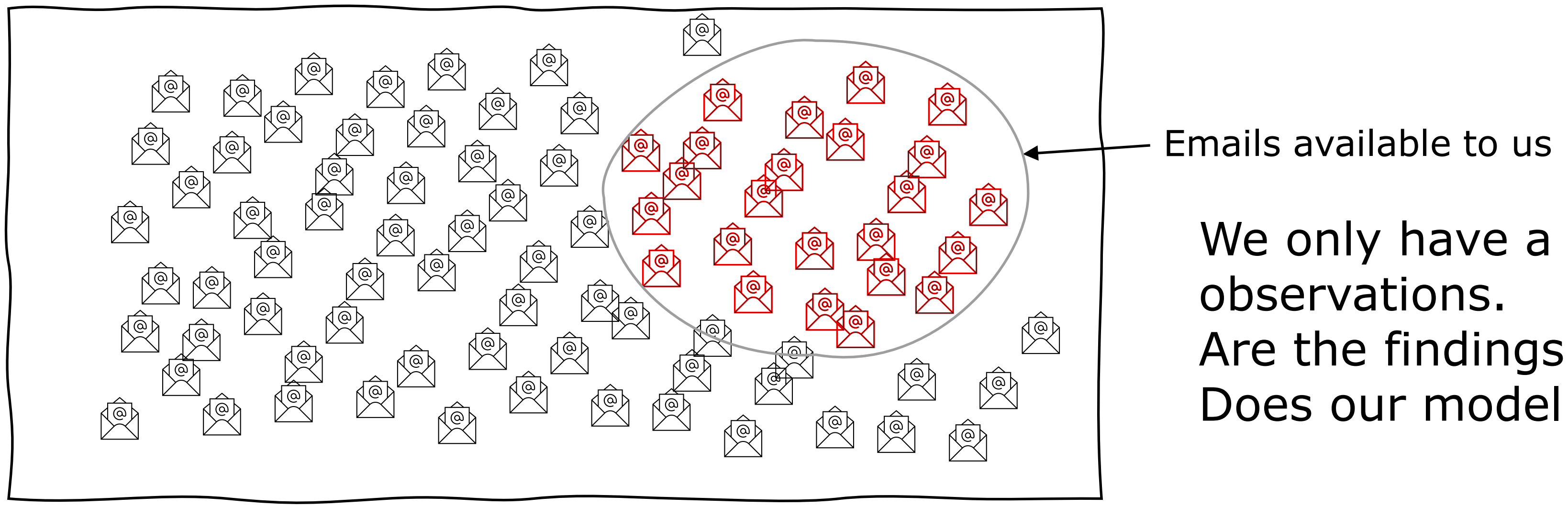
Data splitting

Why is it necessary?



Data splitting

Why is it necessary?



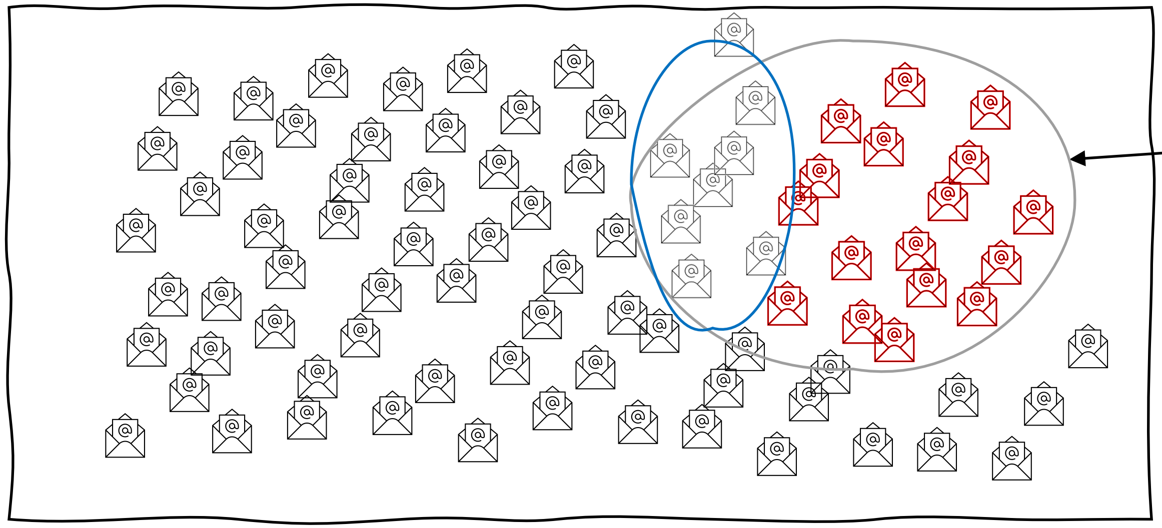
All existing emails sent over the internet

We only have a subset of potential observations.
Are the findings generally valid?
Does our model generalize?

Data splitting

Why is it necessary?

We could „hide" a portion of the initial dataset → Data Splitting



Emails available to us

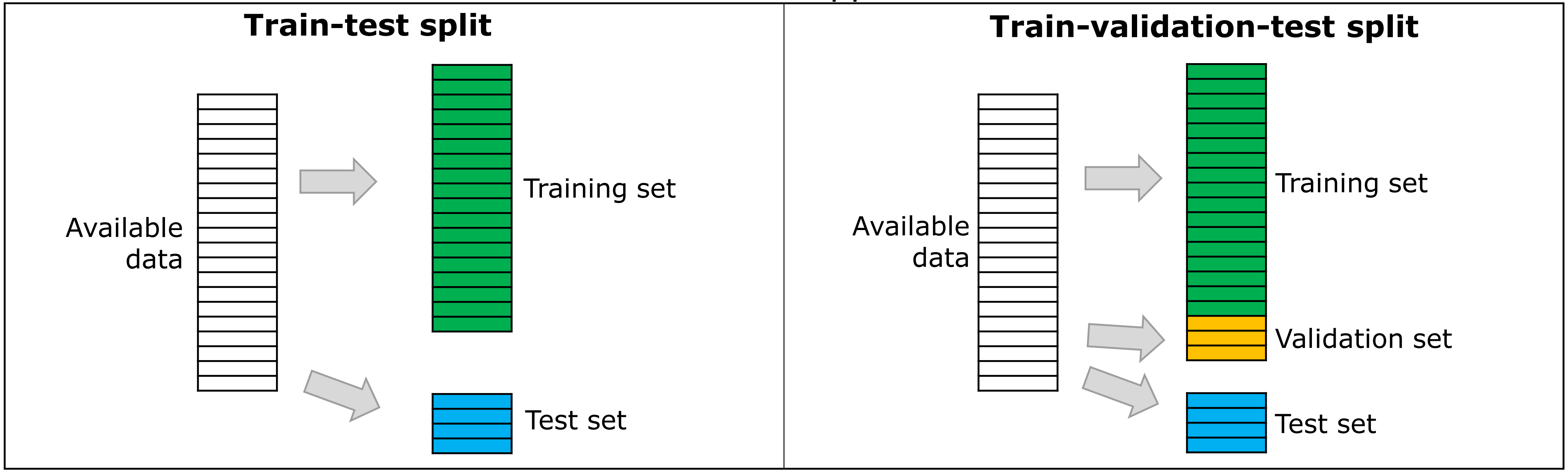
All existing emails sent over the internet

We only have a subset of potential observations.
Are the findings generally valid?
Does our model generalize?

Data splitting

How it's done

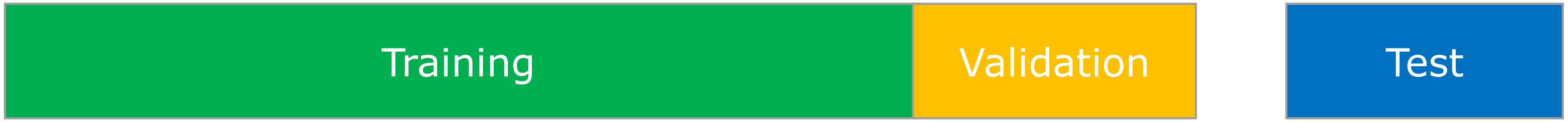
Two common approaches



Data splitting

How it's done

- **Training set**: The subset of the dataset that we use to train the model.
- **Validation set**: The subset of the dataset that we use to tune the model.
- **Test set**: The subset of the dataset that we use to test the final model. HOLD OUT UNTIL THE END



- **Important:**
  - All three sets must have similar characteristics (general trends) of the original dataset.
  - Classes must be properly represented. Statistical properties should be equal.

Data splitting

Choosing the split strategy

- Commonly used split strategies:
  - Random – Choose random samples of the initial dataset
  - Stratified (e.g., in highly unbalanced data) – Choose such that the output class is balanced in the two splits
  - Sequential (e.g., in time series) – Choose depending on the timestamp
- Commonly used splits:
  - Training-Validation-Test: 60/20/20, 50/25/25, 80/10/10
  - Training-Test: 80/20, 75/25, 50/50
- But it depends on the dataset size, the number of features, and the type of problem.
  - Sometime even 99/1 (or 99/0.5/0.5) is OK, e.g., for very large dataset.

Data splitting

Useful high-level functions from the main ML programming languages

- Python
  - *sklearn.model_selection.train_test_split(…)*
  - *sklearn.model_selection.TimeSeriesSplit(…)*
- R
  - *Caret::createDataPartition(…)*
  - *Caret::createTimeSlices(…)*
- Julia
  - *MLDataUtils.splitobs(…)*
  - *MLDataUtils.shuffleobs(…)*
  - *MLDataUtils.stratifiedobs(…)*

Data splitting

- Data splitting:
  - Advantages: Very very simple
  - Disadvantages: It wastes data, which some time is a precious resource. We risk losing important patterns/trends in data set, which in turn increases error.

- **Underfitting** occurs when a model is not able to fit the training data properly.
  - Model too simple or training data not sufficient
- **Overfitting** occurs when a model tries to fit the training data so closely that it does not generalize well to new data
  - Model too complex

Data splitting

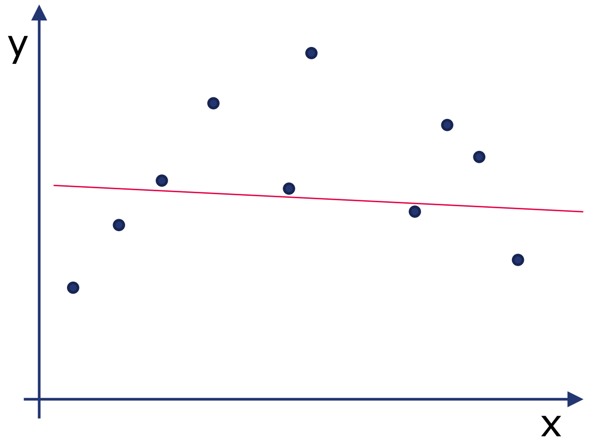
- Warning signs:

Model performs good on the training data and bad on the test data ⇨ Overfitting!

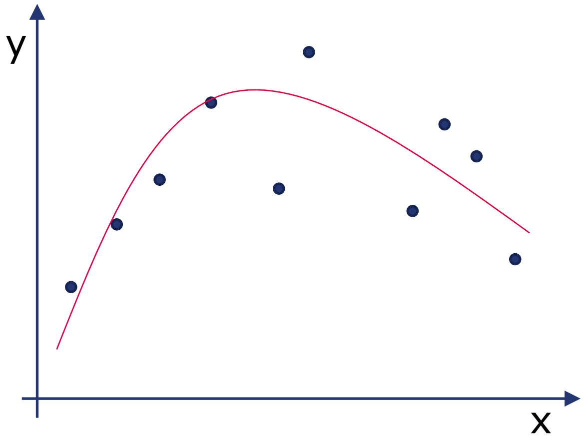
Model performs bad on the training data (and on the test data) ⇨ Underfitting!

Overfitting and underfitting

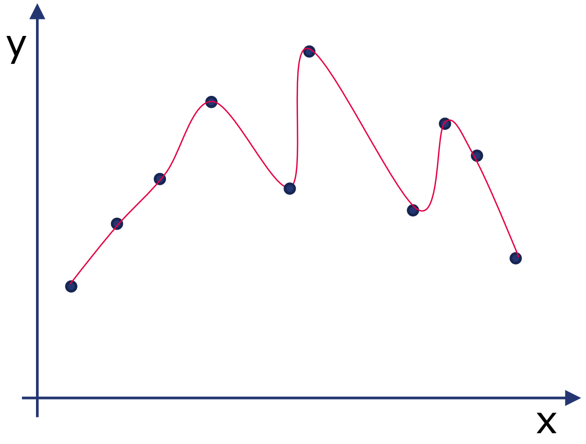
An example with regression



Linear regr. model



quadratic regr. model

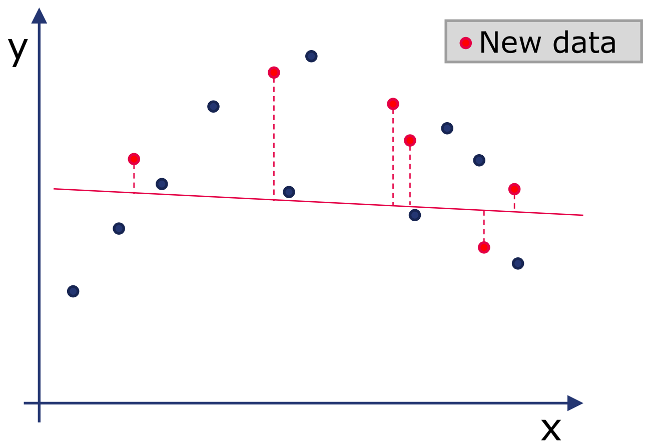


Piecewise non-parametric regr. model

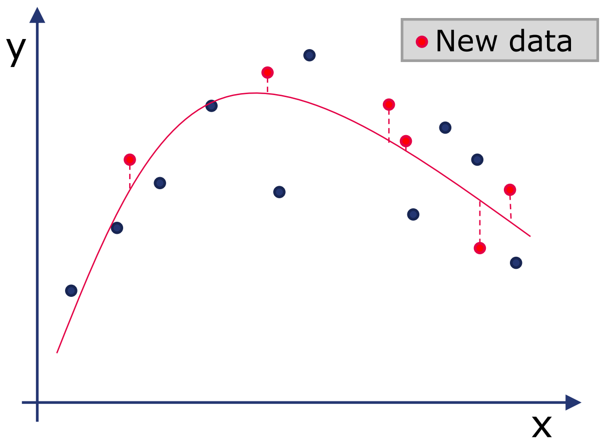
What’s the best model?

Overfitting and underfitting

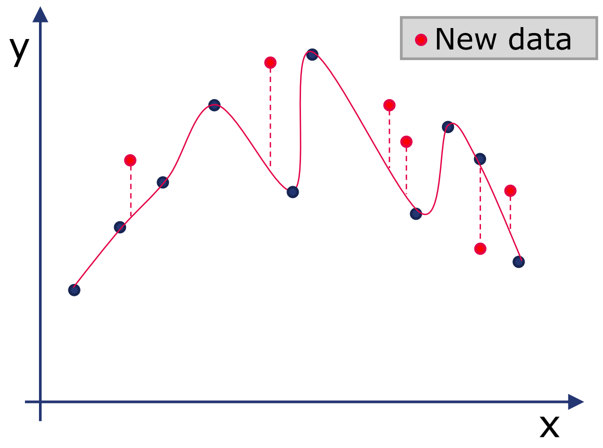
An example with regression



Linear regr. model



quadratic regr. model

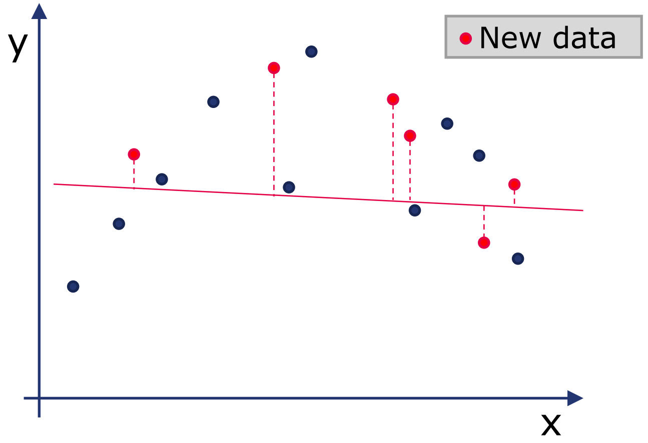


Piecewise non-parametric regr. model

And now?

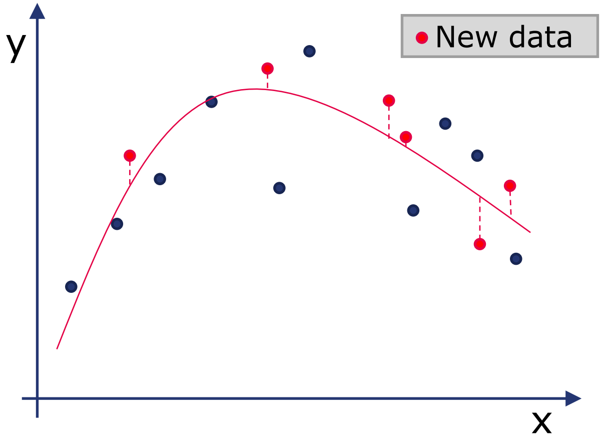
Overfitting and underfitting

An example with regression

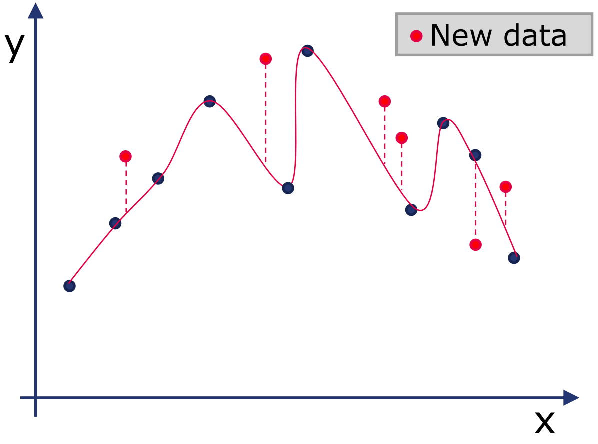


Linear regr. model

UNDERFITTING



quadratic regr. model

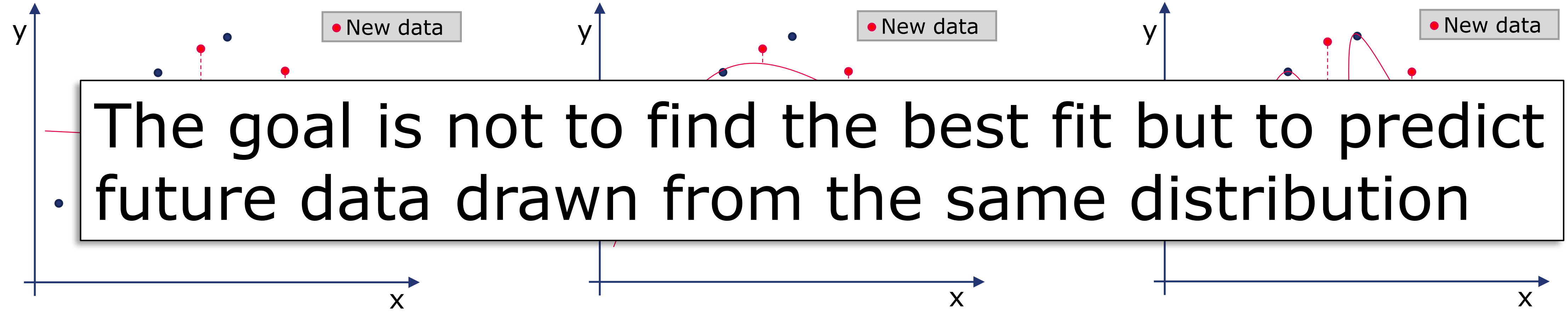


Piecewise non-parametric regr. model

OVERFITTING

Overfitting and underfitting

An example with regression



The goal is not to find the best fit but to predict future data drawn from the same distribution

Linear regr. model

quadratic regr. model

Piecewise non-parametric regr. model

UNDERFITTING

OVERFITTING

Bias and Variance

Definition

The generalization error is the sum of three terms: Bias, Variance, and Noise

- **Bias** – Error caused because the model can not represent the data ("how far we are off the target?)"
  - The model has limited flexibility to learn the true signal from a dataset

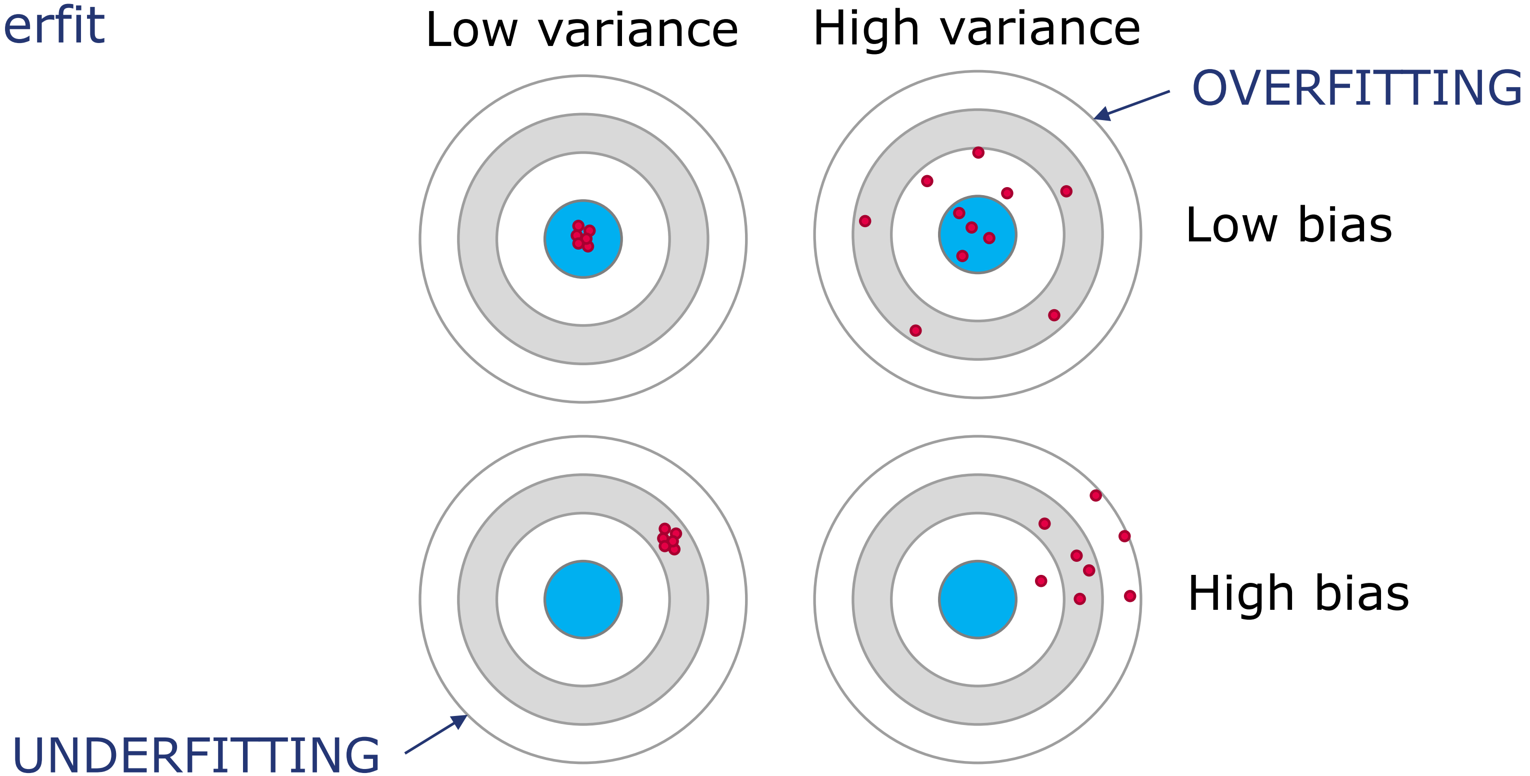
- **Variance** – Error caused because the learning algorithm overreacts to small changes in the training data ("how sparse is the error of our predictions")
  - The model is too sensitive to specific sets of training data

Bias and Variance

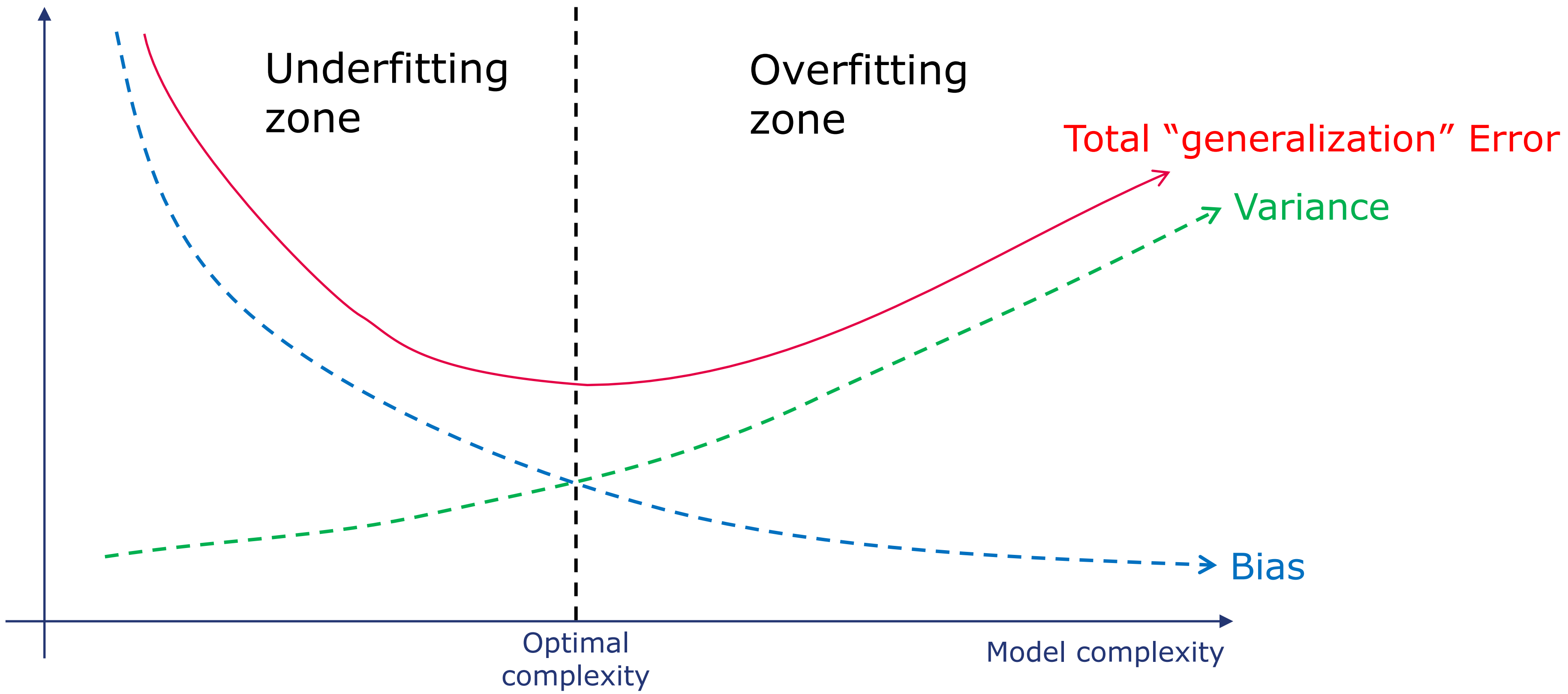
Relation with overfit and underfit

- **High bias, low variance**
  model are consistent but inaccurate on average

- **High Variance, low bias**
  model are accurate on average, but inconsistent.

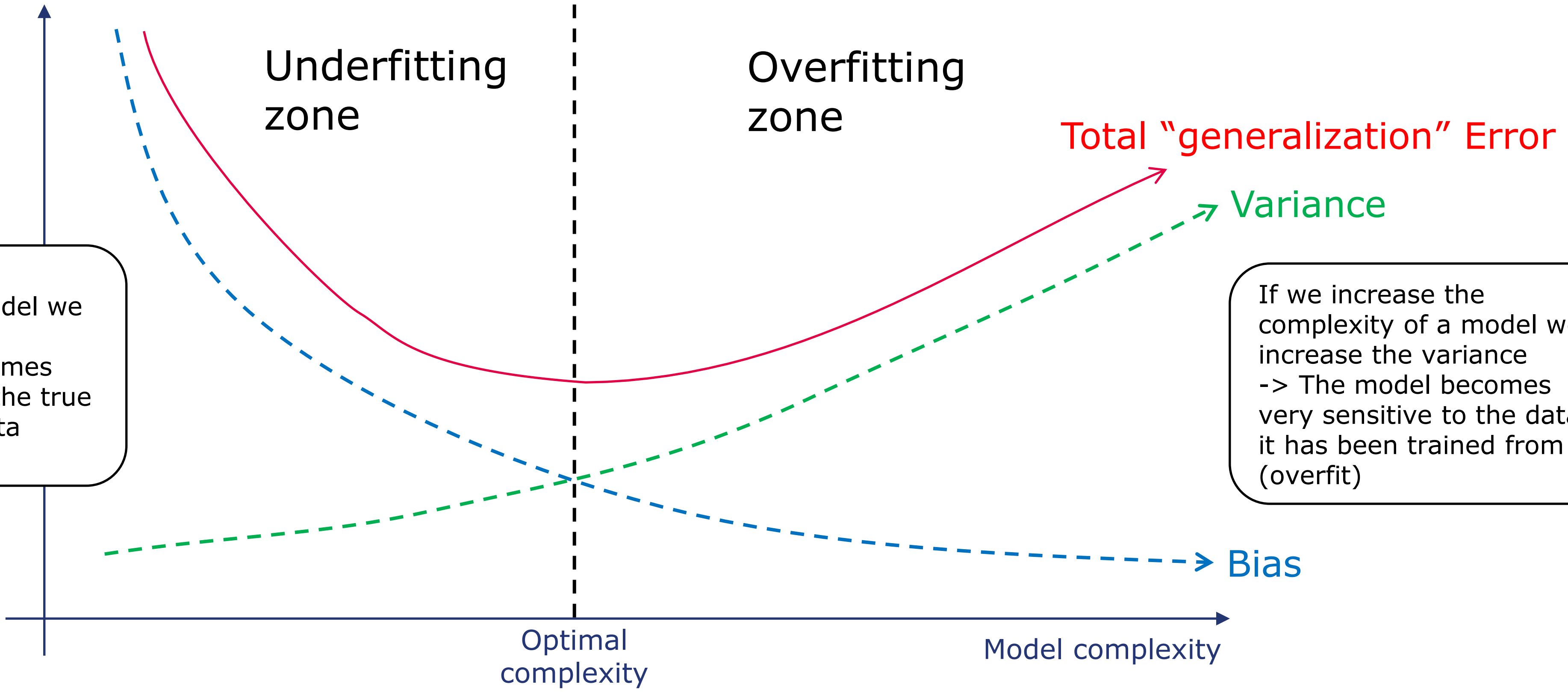


Bias and Variance trade-off



Bias and Variance trade-off

If we decrease the complexity of a model we increase the bias
-> The model becomes unable to capture the true structure of the data (underfit)

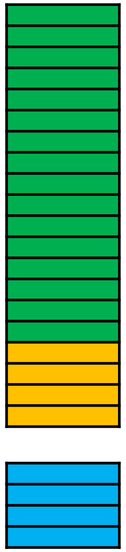


If we increase the complexity of a model we increase the variance
-> The model becomes very sensitive to the data it has been trained from (overfit)

FH KREMS
UNIVERSITY OF APPLIED SCIENCES / AUSTRIA

Cross Validation

Simple split

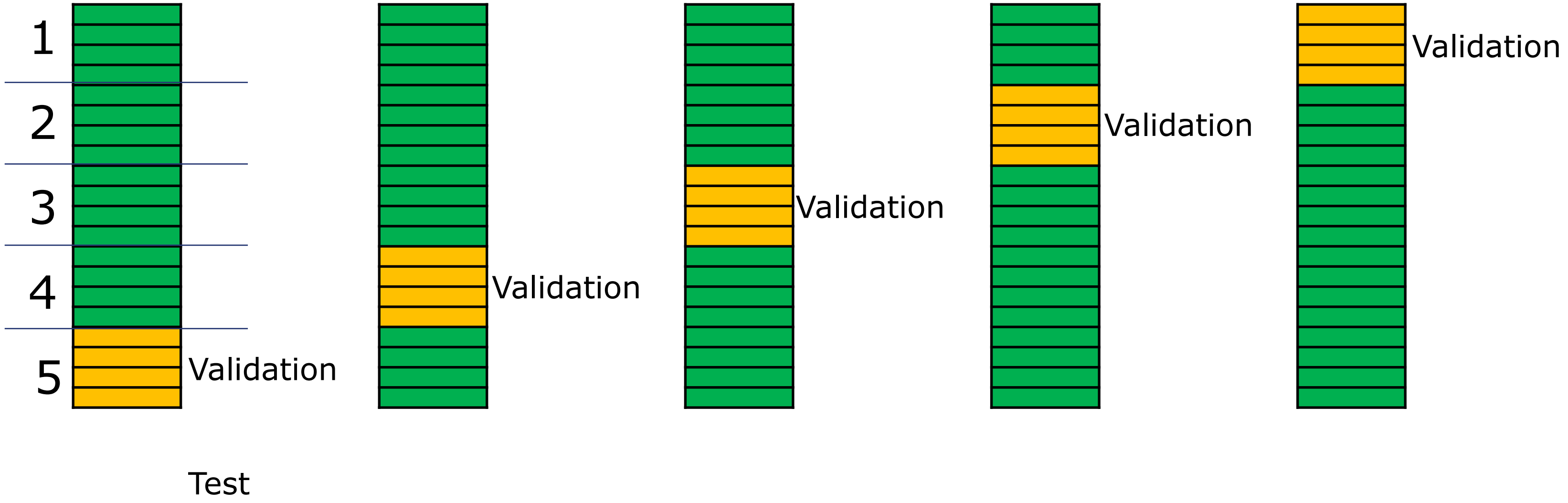


What if the dataset is small?

- It’s a pity to waste so many observations for validation and testing
- Remember that without a proper test set we can’t evaluate effectively our model

Cross Validation

K-fold cross-validation



- Split the dataset *k* times
- For every observation is used:
  - *1* time for testing.
  - *k-1* times for training.
- Compute a metric each time
- Final performance is the average of the k metrics.

Cross Validation

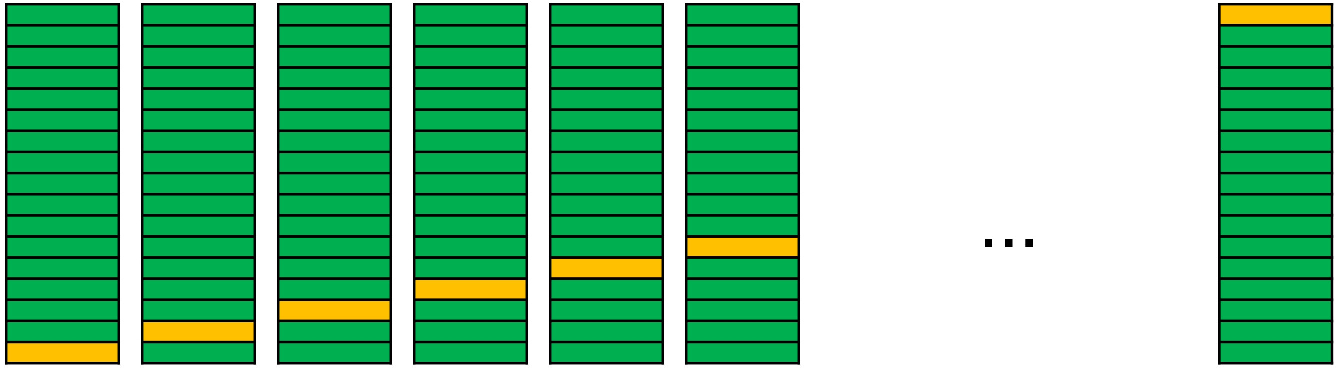
Pseudocode

```
for i in 1…K {
    TrainX, TrainY        = GetAllDataExceptFold(dataX, dataY, i)
    ValidateX, ValidateY = GetDataInFold(dataX, dataY, i)

    # do feature engineering/selection on TrainX, TrainY, ValidateX, and ValidateY

    model.fit(TrainX, TrainY)
    TotalEvaluation += EvaluationMetric(model.predict(ValidateX), ValidateY))
}

FinalEvaluation = TotalEvaluation / K
```

Cross Validation

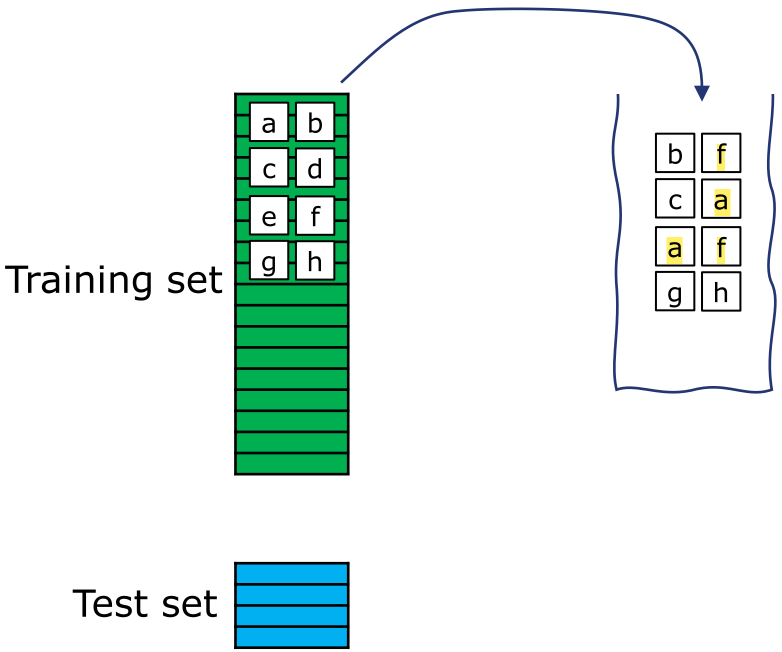
Choice of K

- $K = 5$ *or* 10 ← **Used most of the times**
- $K = N$ ← Special case also called *leave-one-out-validation* (LOOV)



  - It can be used when computationally feasible (small data or simple models)

Bootstrapping and Bagging

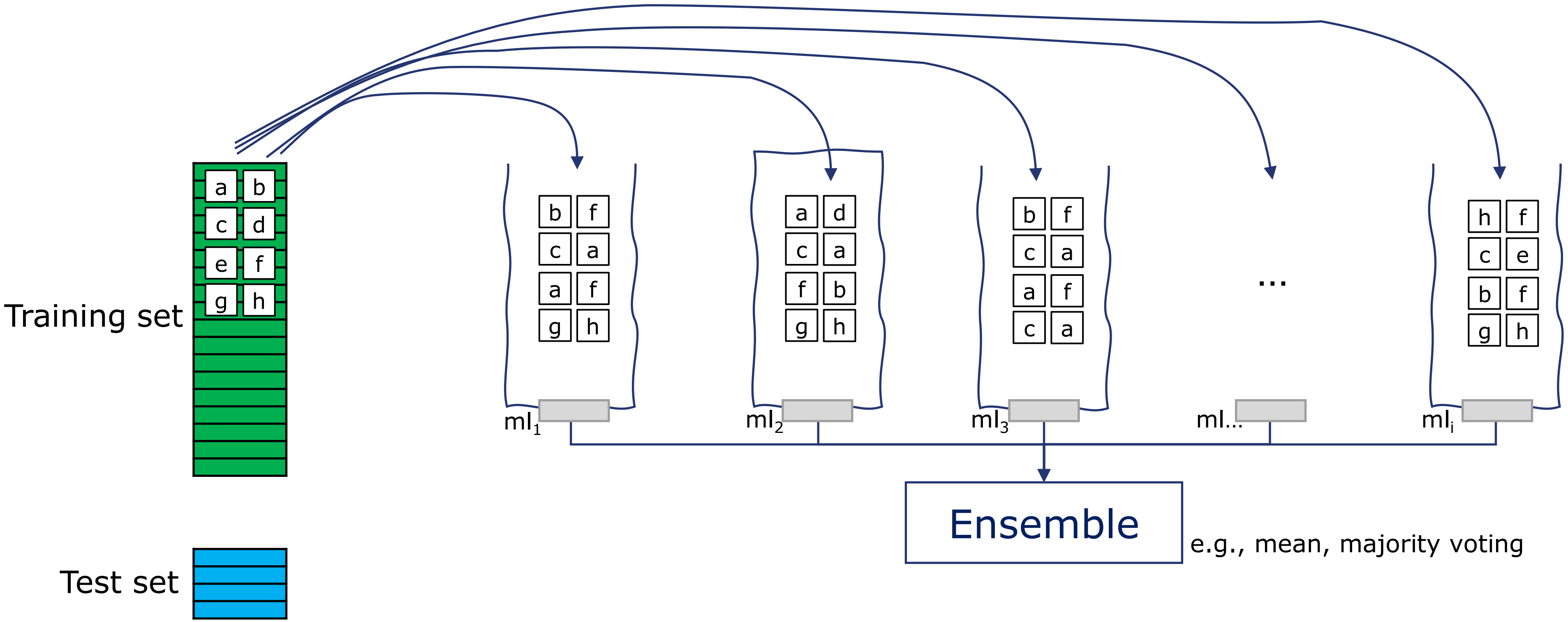


Bootstrapping: Resampling with replacement allows duplicates

The model is trained on a resampling of the training data.

Inference *sample* → *population*
modelled by the inference *resampled* → *sample*.

Bootstrapping and Bagging



Small in-class exercise

Value	Class
1	A
5	A
7	B
…	…

R, Python, or Julia. Pick your favourite.

- Create a data.frame (or data.table) with 1000 observations of 2 columns
    - *value* = random number between 1 and 10
    - *class* = A, B, or C (800x A, 150x B, 50x C)
- Apply a 90/10 Training-testing split
    - By using an own-made function (e.g., by sampling the indexes of the dataframe)
    - By using one of the mentioned functions with stratification.

    → ***PY***: *sklearn.model_selection.train_test_split(…).* → ***R***: *caret::createDataPartition(…).* → ***Julia***: *MLDataUtils.stratifiedobs(…)*
- Observe how the three Classes are distributed in the train and test data sets
- Run multiple times and observe the difference