Introduction to Machine Learning

Coding for Reproducible Research

March 2025

Collaborative doc: https://tinyurl.com/fd3cd22b

Course Leader:

- Simon Kirby

Course Helpers:

- Finley Gibson
- Sam Fletcher

Sign in here

Code of Conduct

- Our ethos is to provide a welcoming and supportive environment for all people, regardless of background or identity. By registering to attend this workshop, participants are agreeing to abide by the Researcher Development Code of Conduct.

- Our goal is to support you to develop your programming skill sets to enable you to do cutting edge research. We want to create a positive and professional learning environment and therefore encourage the following kinds of behaviours:

    - Show courtesy and respect towards all who attend a workshop or engage in community events.
    - Be respectful of different viewpoints and experiences.
    - Gracefully accept constructive criticism.
    - Be patient if there are technical glitches. While we know something about how to use computers, we are not immune to internet or hardware issues.
    - Respect our policy on not recording workshops to protect the nature of the sessions and ensure we are GDPR compliant.

Programme Funding

The CfRR training programme is supported by:

- Research Software Analytics Group
- Institute for Data Science and Artificial Intelligence (IDSAI)
- University of Exeter Reproducibility Leadership Team
- EPSRC Research Software Engineering Fellowship
- Community of academics who volunteer their time to support delivery

To make the case for continued investment, please help us demonstrate the impact of these sessions by attending all courses you register for and providing feedback at the end of the course.

Intro to Machine Learning

Part 2 - Model selection and evaluation

Course contents

Session 1

- Slides: what is machine learning?
- Tutorial: linear regression
- **Slides: model selection and evaluation**

Session 2

- Tutorial: model selection and evaluation
- Slides: the machine learning pipeline
- Tutorial: machine learning pipeline task

Session 3

- Continue with machine learning pipeline task
- Tutorial: unsupervised learning

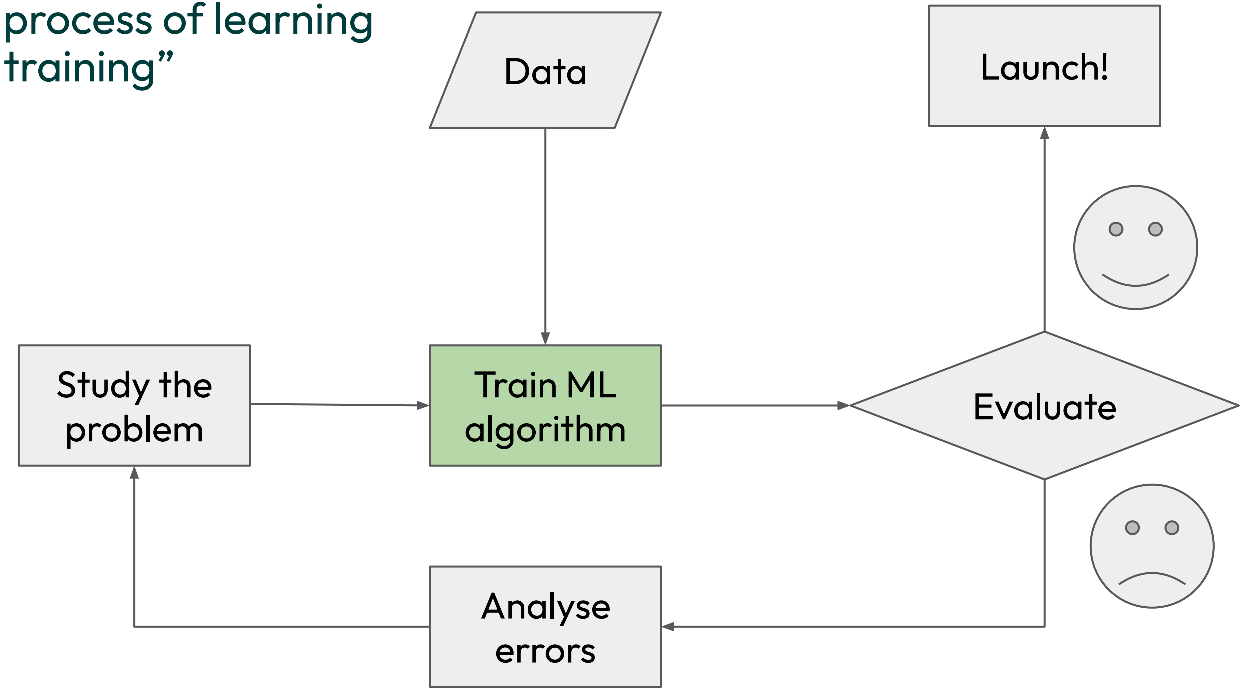
How do we get the best possible model performance?

- Model training - what is happening?
- Model selection - select appropriate model
- Model validation - assess generalisability & prevent overfitting
- Model evaluation - assess model performance

We are going to discuss these four things.

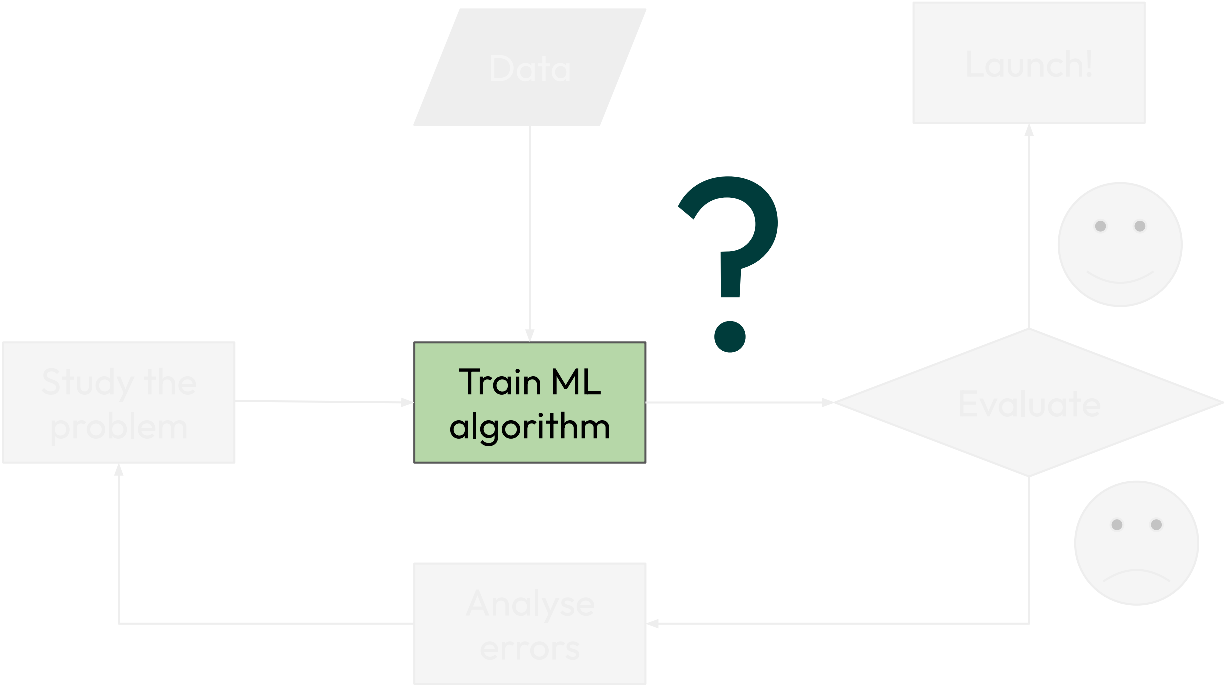
Training a model

We call this process of learning from data "training"

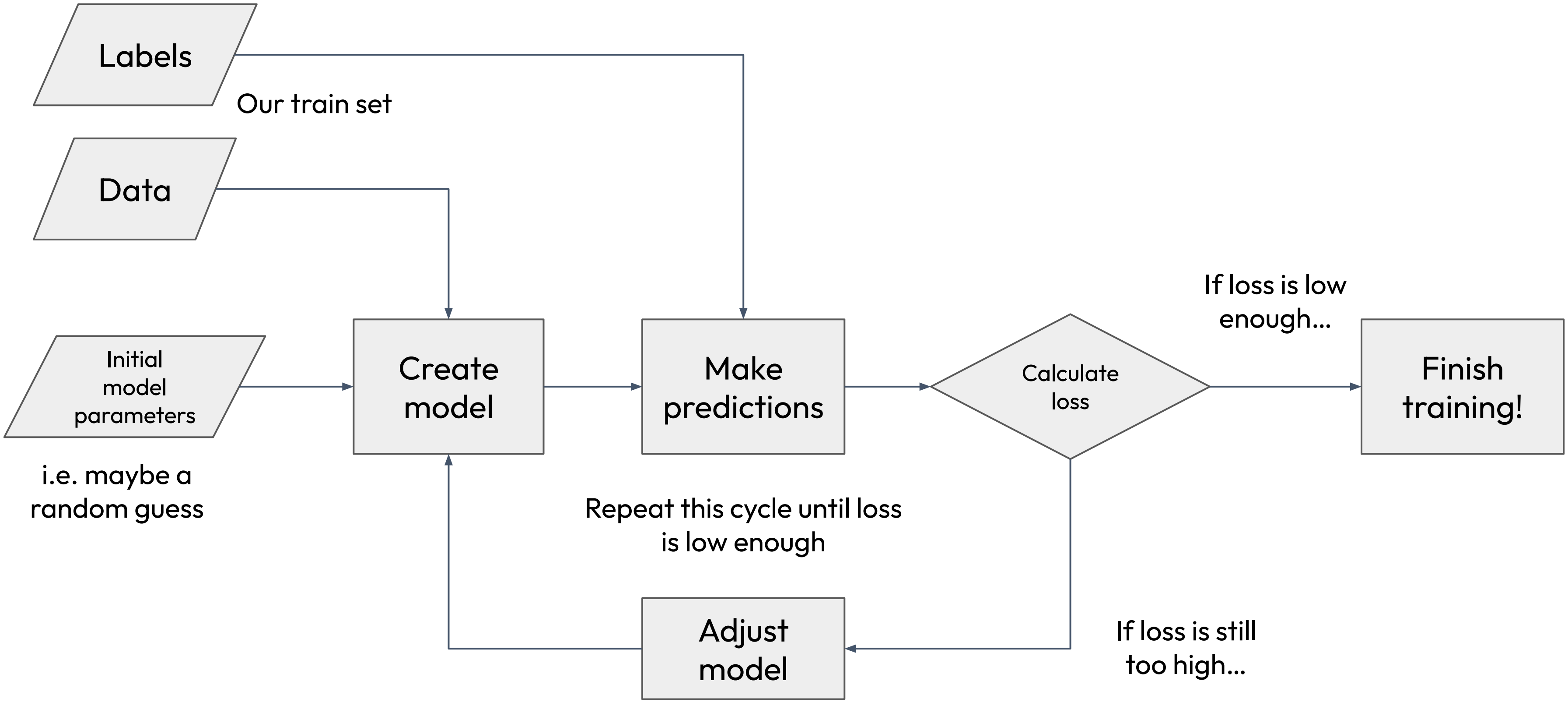


Diagrams adapted from Hands On Machine Learning with scikit-learn, Keras and Tensorflow by Aurelien Geron

Training a model



Training a model



Model selection

- There are different types of machine learning problem
- These will influence what models are appropriate to select
- Hence, first stage of machine learning task is to explore your data

The "no free lunch theorem": Wolpert and Macready

- Applied to machine learning: no single best algorithm for predictive modeling problems, i.e. classification and regression.
- Means you cannot blindly take a "good" algorithm, and expect it to perform well on a new problem.

Model selection - a warning

- Libraries with consistent APIs, such as scikit-learn, make it trivially easy to select different machine learning algorithms, and apply them.
- This can be dangerous: it is possible to select a model that is not appropriate for your use case, or that does not make mathematical or physical sense.
- Similarly, when tuning parameters, these should also be assessed in a similar way.

Model evaluation

- Assess how well the model performs
- Metrics such as:
    - Accuracy (classification)
    - Precision, recall, f1-score (classification)
    - MSE/RMSE (regression)
- Usually assessed on a test (an unseen) dataset
- How do we do this so we are not “marking our own homework?”

Model evaluation - train test split

First thing we should always do is split our data

- Train set: the data we use for training our model - commonly 80%
- Test set: data we evaluate our model on/assess its performance on - 20%
- Should be before you perform serious investigation into the data!

Commonly 80% 20%

Train test split: bias

It is really common to hear about problems of bias in ML algorithms.

Bias can creep in at this stage!

- Your test set needs to be representative of the training data (and vice versa)
- Assessing bias in these splits will also require domain/problem knowledge!

- i.e. if your data is sorted by gender, dont take the last 20% as your test set
    - Because your train set could be mostly the other gender
- We can use scikit-learn to do lots of this for us.

Model validation: overfitting vs underfitting

Overfitting: learning the training features.

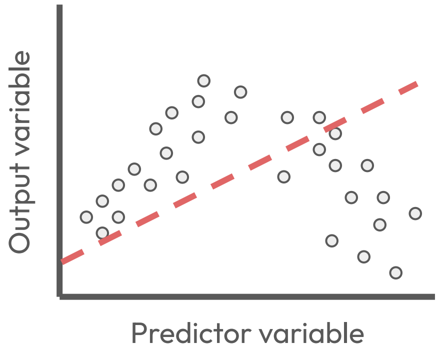
- Symptoms: high accuracy on train set, low accuracy on test set

Underfitting: model has not/cannot learnt the training features

- Symptoms: low accuracy on train AND test sets

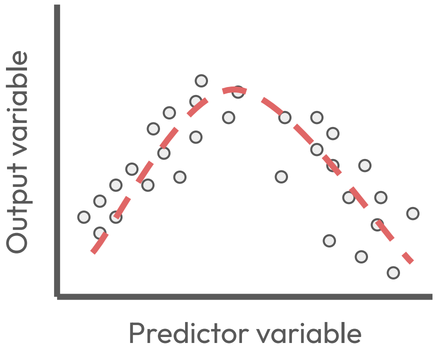
Model validation: overfitting vs underfitting

This can be visualised with a polynomial with different degrees of freedom



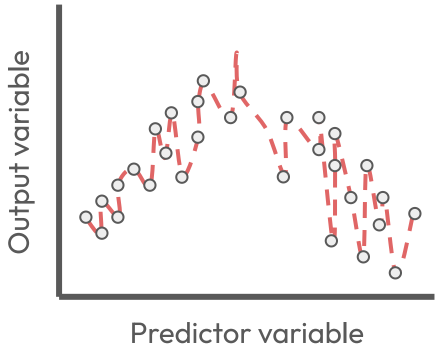
Model is underfitting:

Poor accuracy, but not possible to learn the shape of the data



Model is fitting well:

Good accuracy: general shape learnt well

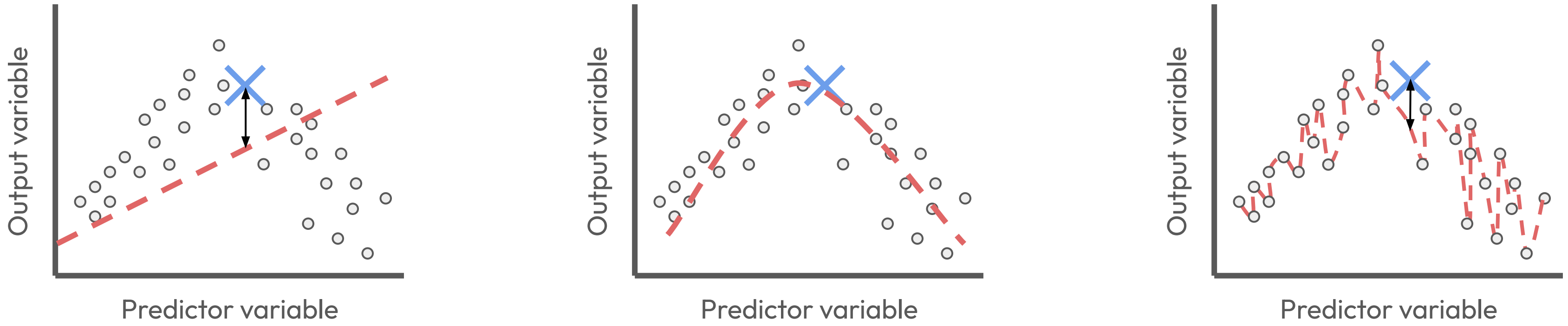


Model is overfitting:

Perfect accuracy on this data. **So what is the problem?**

Model validation: overfitting vs underfitting

Lets add a new, unseen point. This could be from the test set.



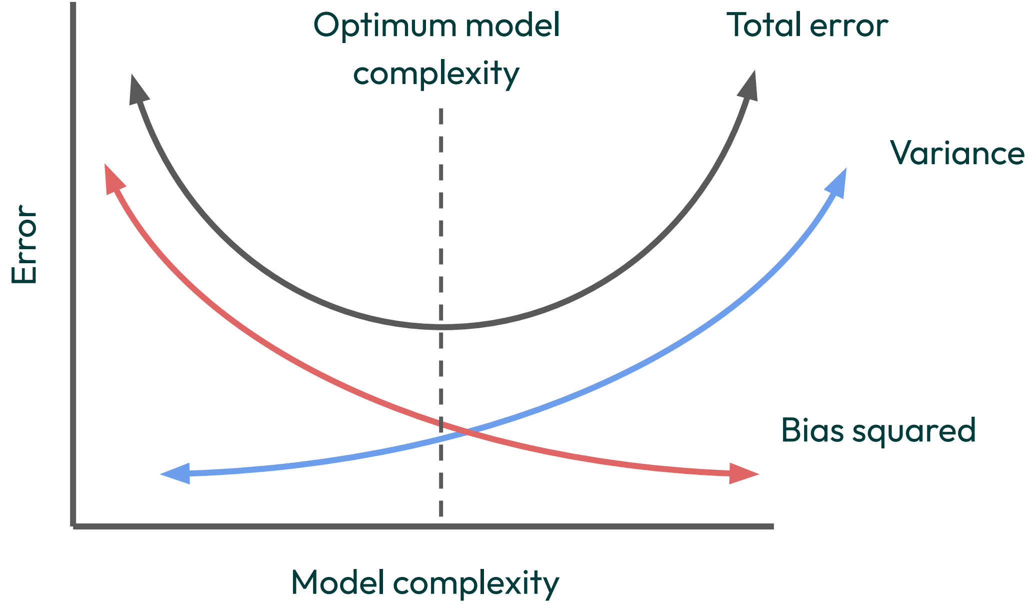
Model does not predict new point well

Model predicts new point well

Model does not predict new point well

Model validation: bias variance tradeoff

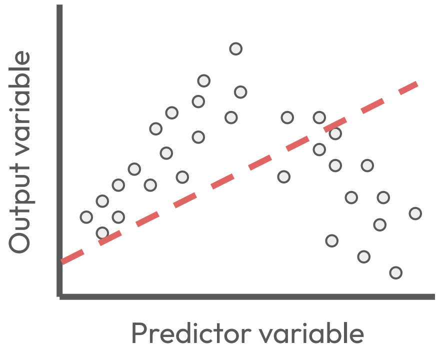
Total error is a function of the bias and the variance



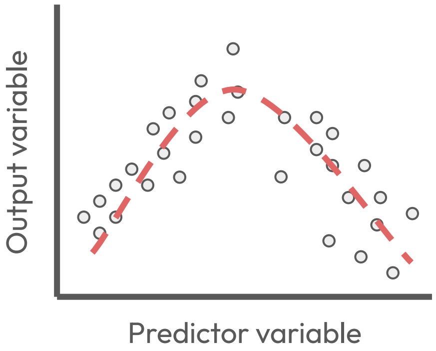
Model validation: bias variance tradeoff

University of Exeter

Total error is a function of the bias and the variance

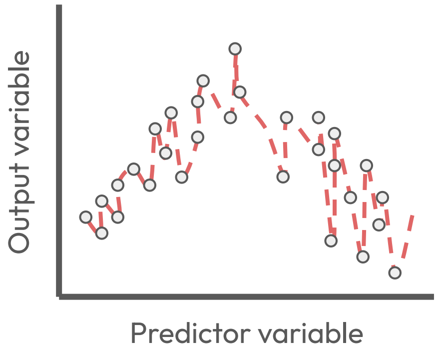


Low complexity, high bias, low variance



Optimal complexity, minimises error

Optimal balance of bias and variance



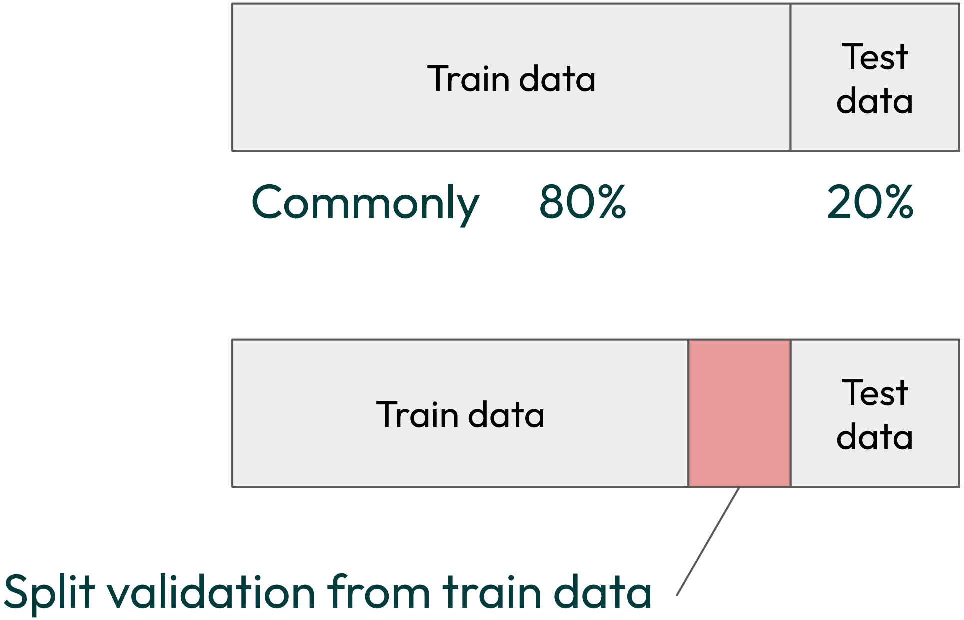
High complexity, low bias, high variance

Model validation: in summary

- A model that is generalisable makes accurate predictions on new, unseen data.
- Models that do not generalise well have not learned a true relationship between the input features, and the outcomes.
- A model that has been overfit is often not generalisable.

Better evaluation: validation holdout

During training, we can split our training set up into a train set and a validation set



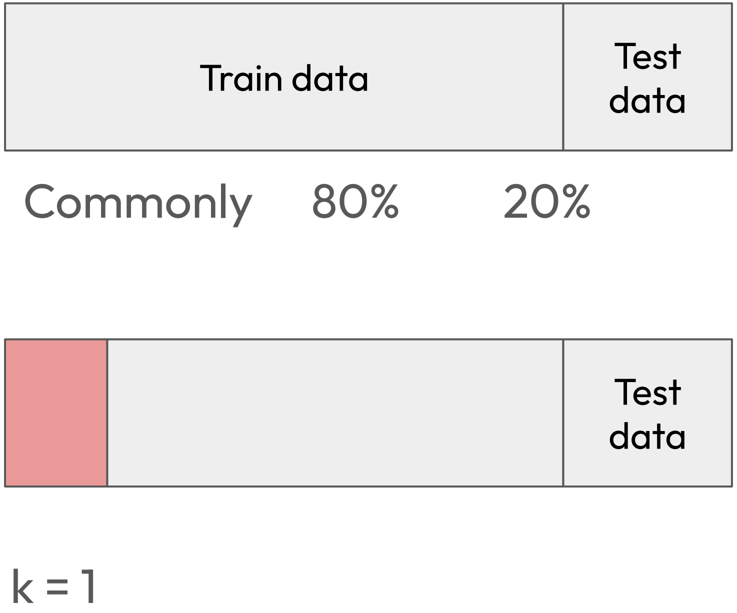
This is a good way of preventing overfitting.

Introduces a problem: reduces the size of our train set.

Better evaluation: validation holdout

University of Exeter

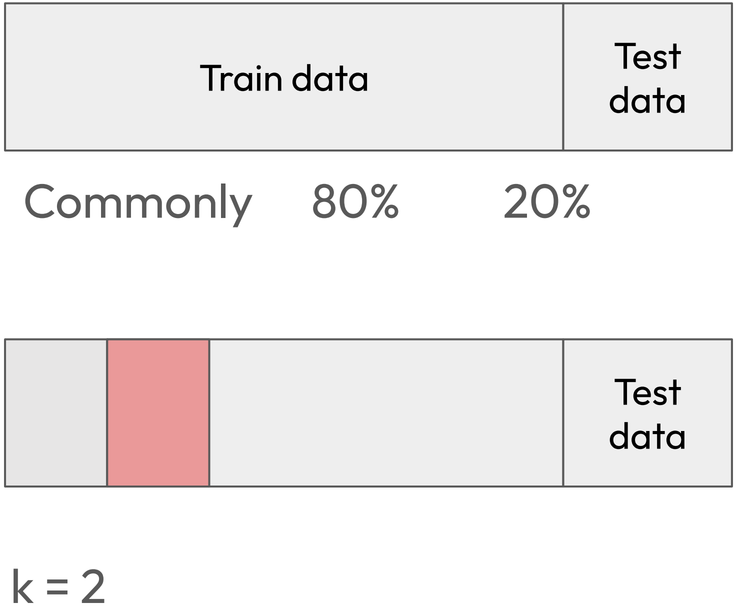
We can do this k times: k-fold cross validation. Allows us to still use this validation data in training.



We assess the performance on each “fold”, helping to reduce change of overfitting

Better evaluation: validation holdout

We can do this k times: k-fold cross validation. Allows us to still use this validation data in training.

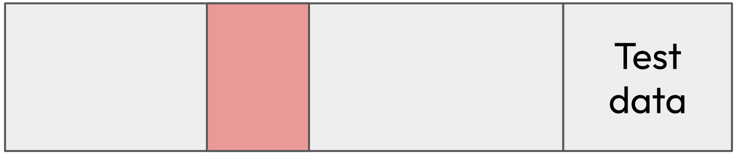


We assess the performance on each “fold”, helping to reduce change of overfitting

Better evaluation: validation holdout

We can do this k times: k-fold cross validation. Allows us to still use this validation data in training.

Commonly 80% 20%

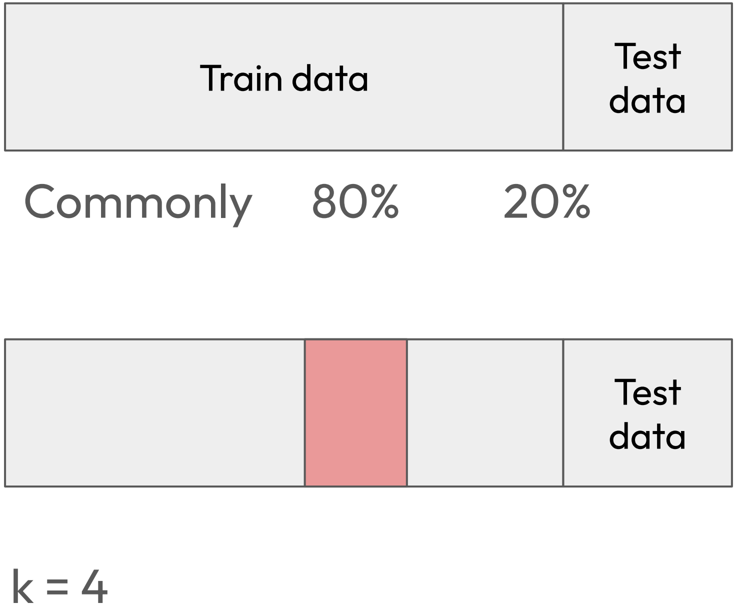


k = 3

We assess the performance on each “fold”, helping to reduce change of overfitting

Better evaluation: validation holdout

We can do this k times: k-fold cross validation. Allows us to still use this validation data in training.



We assess the performance on each “fold”, helping to reduce change of overfitting

Thank you!

University of Exeter

Post-workshop Anonymous Feedback Form 2024-25

https://tinyurl.com/2d8fys7e