

Training, validating and testing

Herman Kamper

2024-01, CC BY-SA 4.0

How do we know which model to pick?

E.g. for linear regression we can choose:

- The degree of the polynomial features.
- The number of RBFs.
- Whether to use regularisation.
- The value of λ for L_1 or L_2 regularisation.
- Etc.

How do we choose which of these options to use?

We could look at the loss (or another metric such as MSE or RMSE) on the training data.

But this is problematic since the more complex/expressive a model is (e.g. higher-order polynomial, lower λ), the better it will do on the training data.

Idea 1: Training and validation sets

We define a held-out validation set which we don't train on.

We then evaluate different model options only on the validation set.

$(\mathbf{x}^{(1)}, y^{(1)})$, $(\mathbf{x}^{(2)}, y^{(2)})$, $(\mathbf{x}^{(3)}, y^{(3)})$, $(\mathbf{x}^{(4)}, y^{(4)})$, \dots , $(\mathbf{x}^{(N)}, y^{(N)})$

Idea 2: Training, validation and test sets

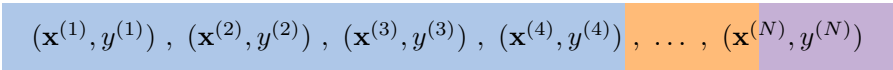
The problem with just training and validation sets

If we follow the idea from above, how well would you expect the model to do on completely new data? Could you tell just from the validation set performance?

We could report metrics on the validation set, but this will likely be optimistic (because we chose hyperparameters like λ to be optimal on the validation set).

A test set

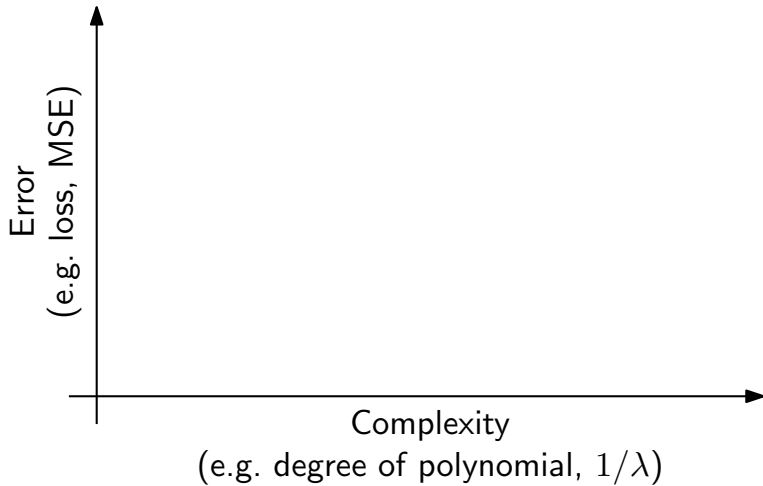
We report final performance on a completely held-out test set, which doesn't overlap with either the training or validation sets:



$(\mathbf{x}^{(1)}, y^{(1)}) , (\mathbf{x}^{(2)}, y^{(2)}) , (\mathbf{x}^{(3)}, y^{(3)}) , (\mathbf{x}^{(4)}, y^{(4)}) , \dots , (\mathbf{x}^{(N)}, y^{(N)})$

You should use this set as little as possible (ideally never) while developing your model.

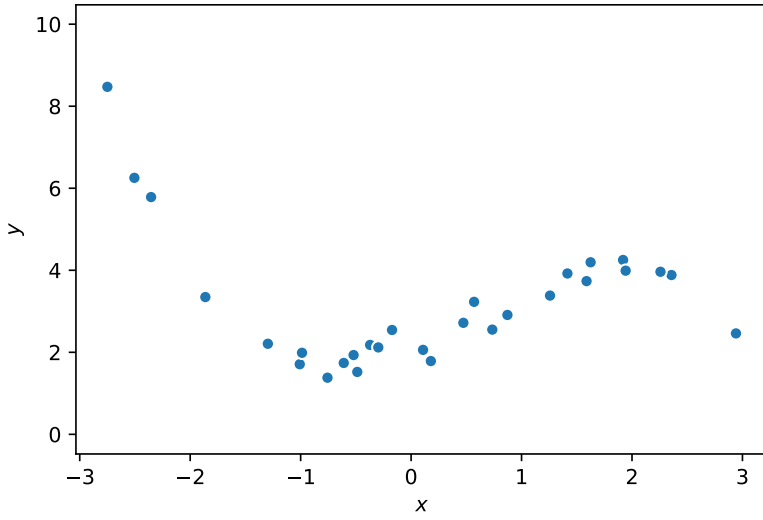
What does over- and underfitting look like?

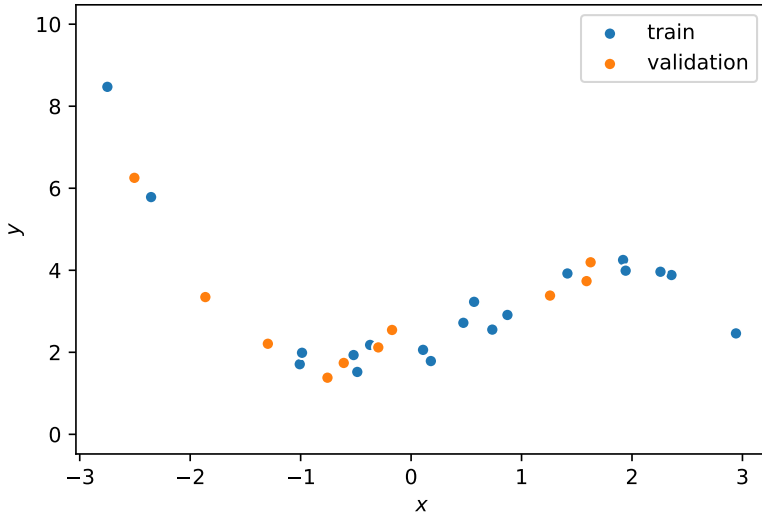


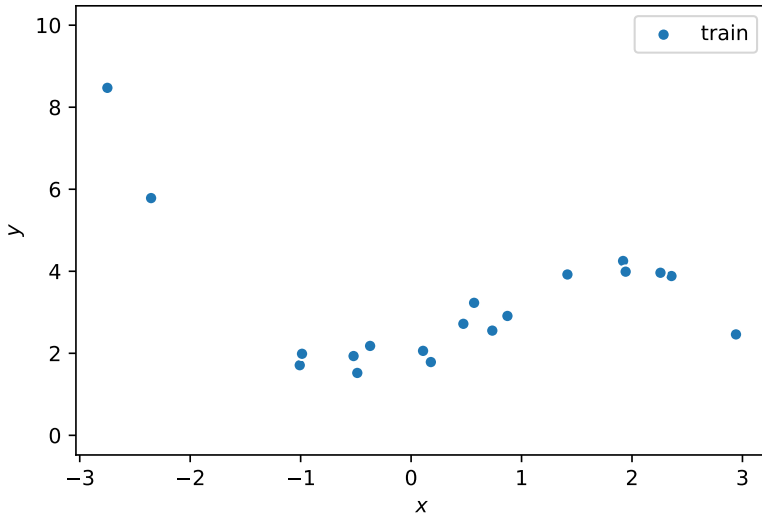
When reporting training loss on curves such as these, we do not include the regularisation term so that the training loss can be compared to the validation loss (where the regularisation loss is never included).

So, which model should we use?

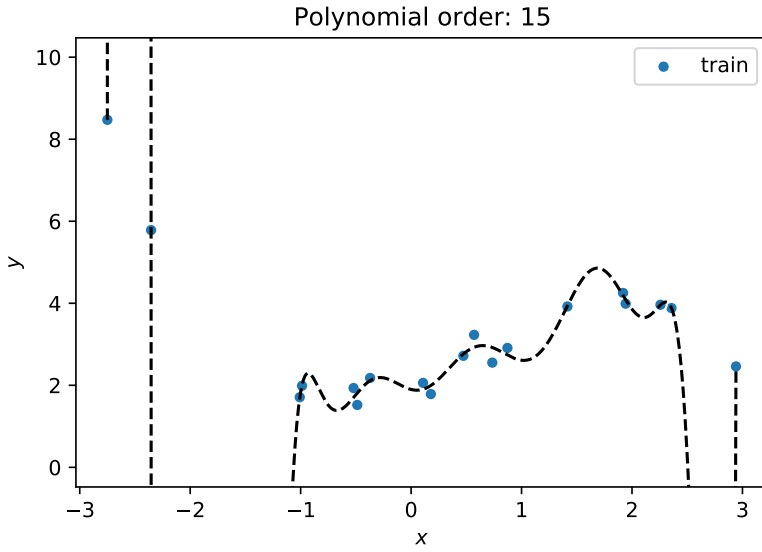
Let's look at an example on the following dataset.



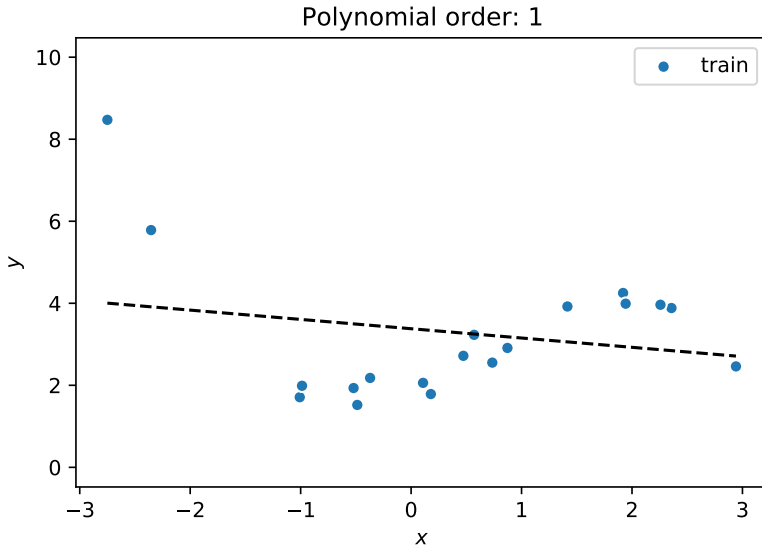




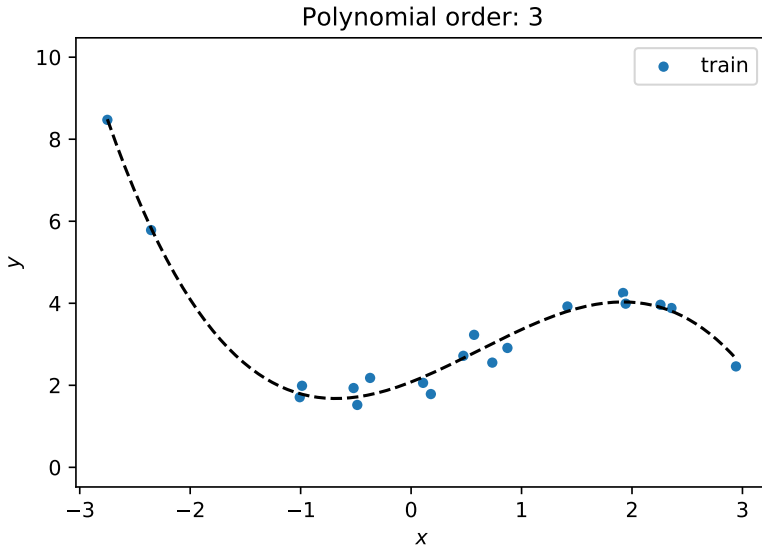
Let's fit some models to the training data.



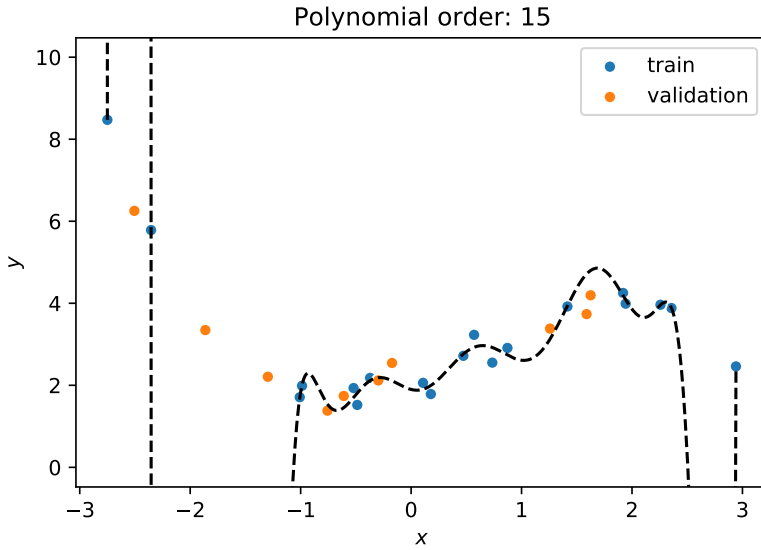
$$\text{MSE}_{\text{train}} = 0.0243$$



$$\text{MSE}_{\text{train}} = 2.5005$$

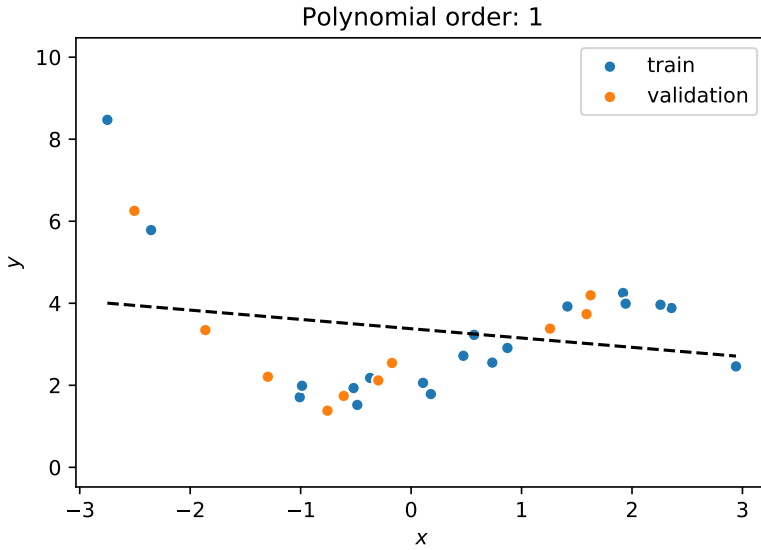


$$\text{MSE}_{\text{train}} = 0.0600$$



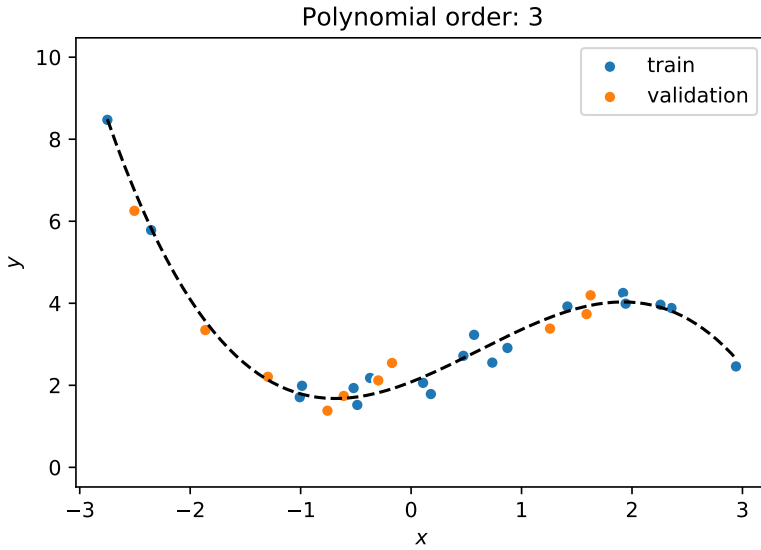
$MSE_{\text{train}} = 0.0243$; $MSE_{\text{val}} = 456386.9249$

(overfitting, "high variance")



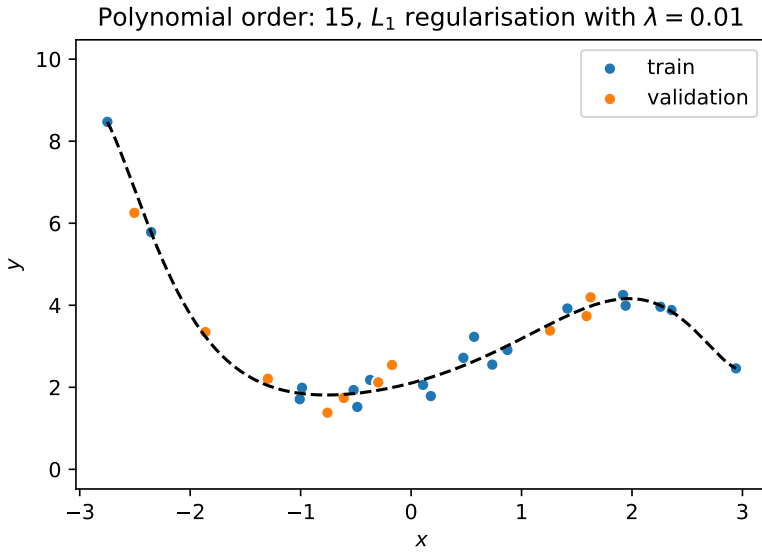
$MSE_{\text{train}} = 2.5005$; $MSE_{\text{val}} = 2.0069$

(underfitting, "high bias")



$MSE_{\text{train}} = 0.0600$; $MSE_{\text{val}} = 0.1056$

("just right")



$MSE_{\text{train}} = 0.0508$; $MSE_{\text{val}} = 0.1039$

Training on test data is one of the worst mistakes you can make in machine learning

Videos covered in this note

- [Training, validating and testing](#) (18 min)

Reading

- ISLR 2.2.1
- ISLR 5.1.1