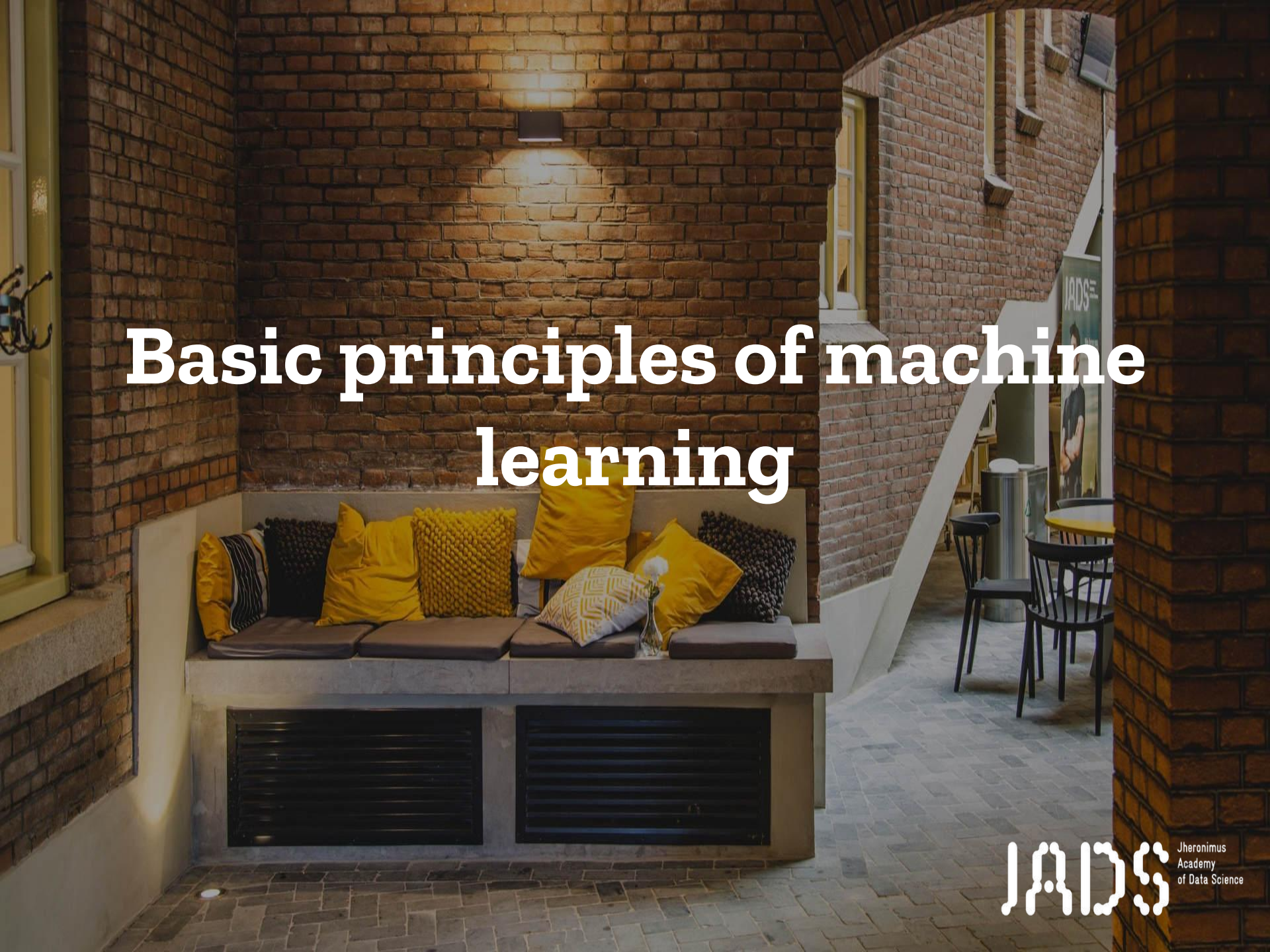

Basic principles of machine learning
JADS
Jheronimus Academy of Data Science

Machine learning

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

Main take away

A machine learning model needs to work well on new data, otherwise it is useless (it needs to be generalizable)

3. IT'S GENERALIZATION THAT COUNTS

The fundamental goal of machine learning is to *generalize* beyond the examples in the training set. This is because, no matter how much data we have, it is very unlikely that we will see those exact examples again at test time. (No-

Pedro Domingos: A Few useful Things to Know about Machine Learning

The essence of the problem

Finding a good relation between input (X's) and output (Y) for the data you have

≠

Finding a good relation between input (X's) and output (Y) for the data you do not have yet

Agenda

1) Overfitting
2) Training error versus test error
3) Bias-variance trade-off
4) Cross validation

⇒ These are the core principles that lie at the heart of every supervised machine learning problem

Overfitting

Overfitting

Wikipedia: *"the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably"*

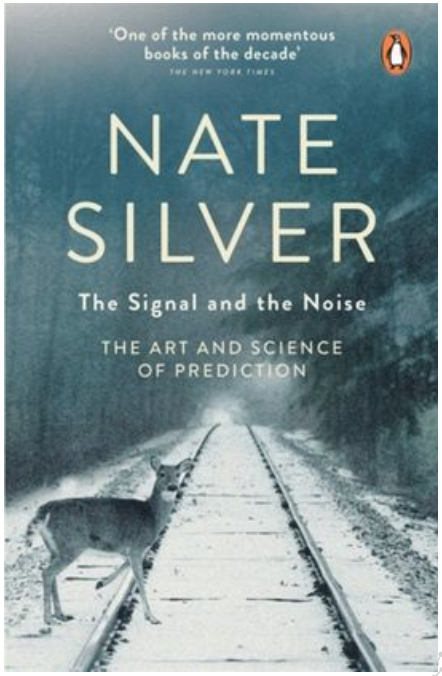
A machine learning model should work well on new data

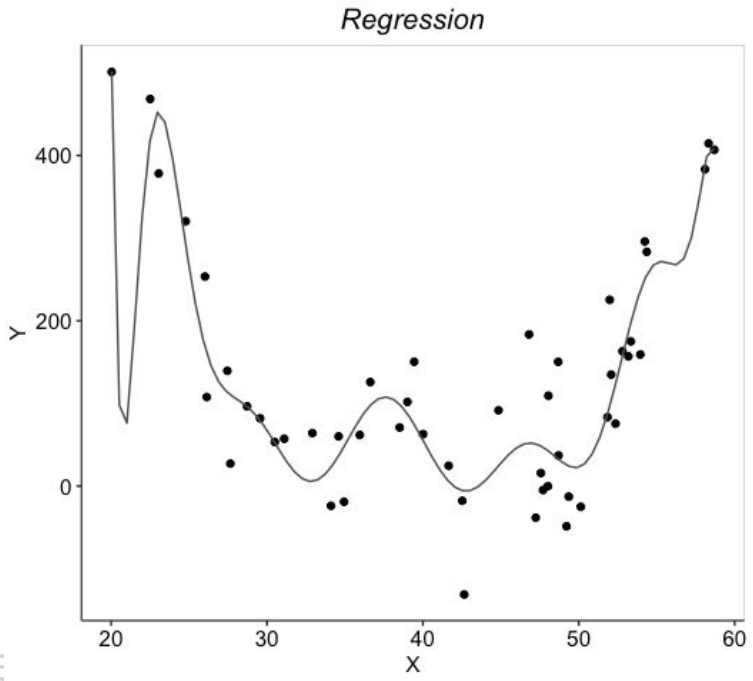
A machine learning model should not suffer from overfitting

Overfitting

Overfitting is the process of modelling the ‘noise’ in your data, on top of the ‘signal’

In other words, catching the particularities in your model, on top of the true relation between input and output





Overfitting

Overfitting is the process of modelling the ‘noise’ in your data, on top of the ‘signal’

In other words, catching the particularities in your model, on top of the true relation between input and output

Which means:

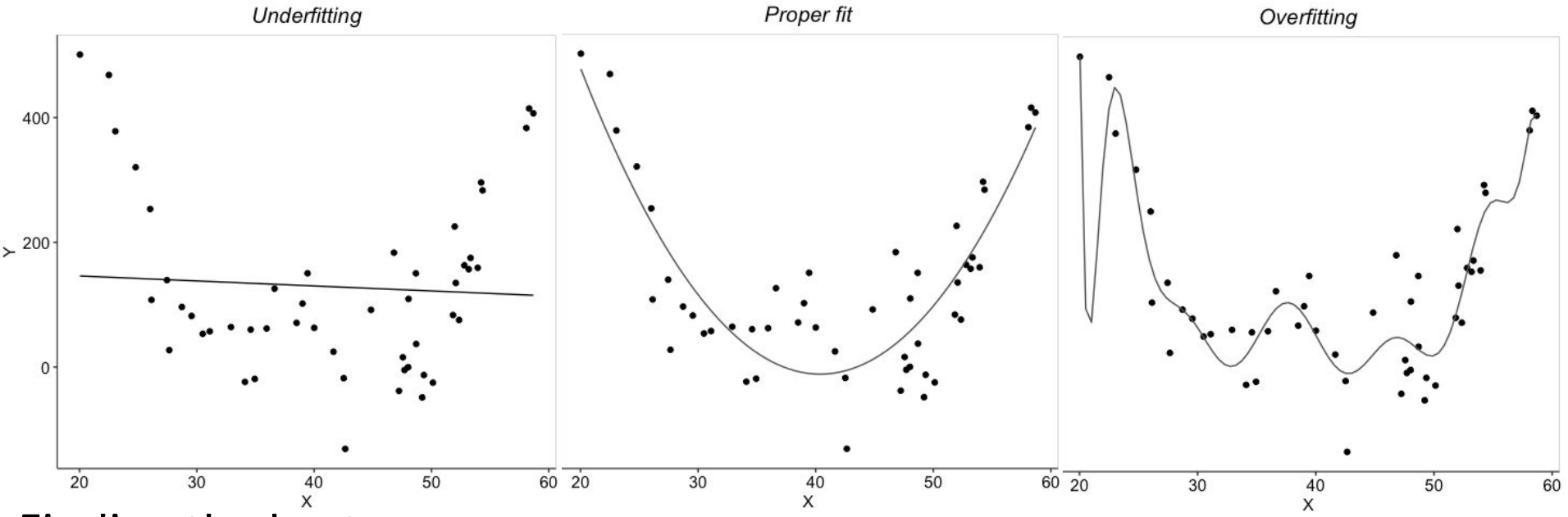
- A ‘good fit’ on the data on which the model was trained
- A ‘poor fit’ when applied to new data

=> Weak generalizability

Overfitting versus underfitting

Overfitting is about modelling ‘noise’, on top of the ‘signal’

Underfitting is about missing part of the signal



Finding the best generalizable model is a search for the optimal level of flexibility / complexity

Glossary

Overfitting: modelling noise, on top of the signal in the data;

Underfitting: missing part of the signal in the data;

Signal: the true relation between input (X's) and output (Y)

Noise: random irregularities in the data, causing variation in Y that has nothing to do with the X's

Machine learning algorithm: model relating input (X's) to output (Y) (in case of supervised machine learning)

Model fit: degree to which the model outcomes resemble the data

Flexibility / complexity: refers to the capability of a machine learning algorithm to capture the signal in the data

Examples of machine learning algorithms

Regression

Gradient Boosting

Logistic regression

LASSO

Support Vector Machine

Neural network

K nearest neighbour

Random Forest

Decision Tree

Naive Bayes classifier

Examples of machine learning algorithms

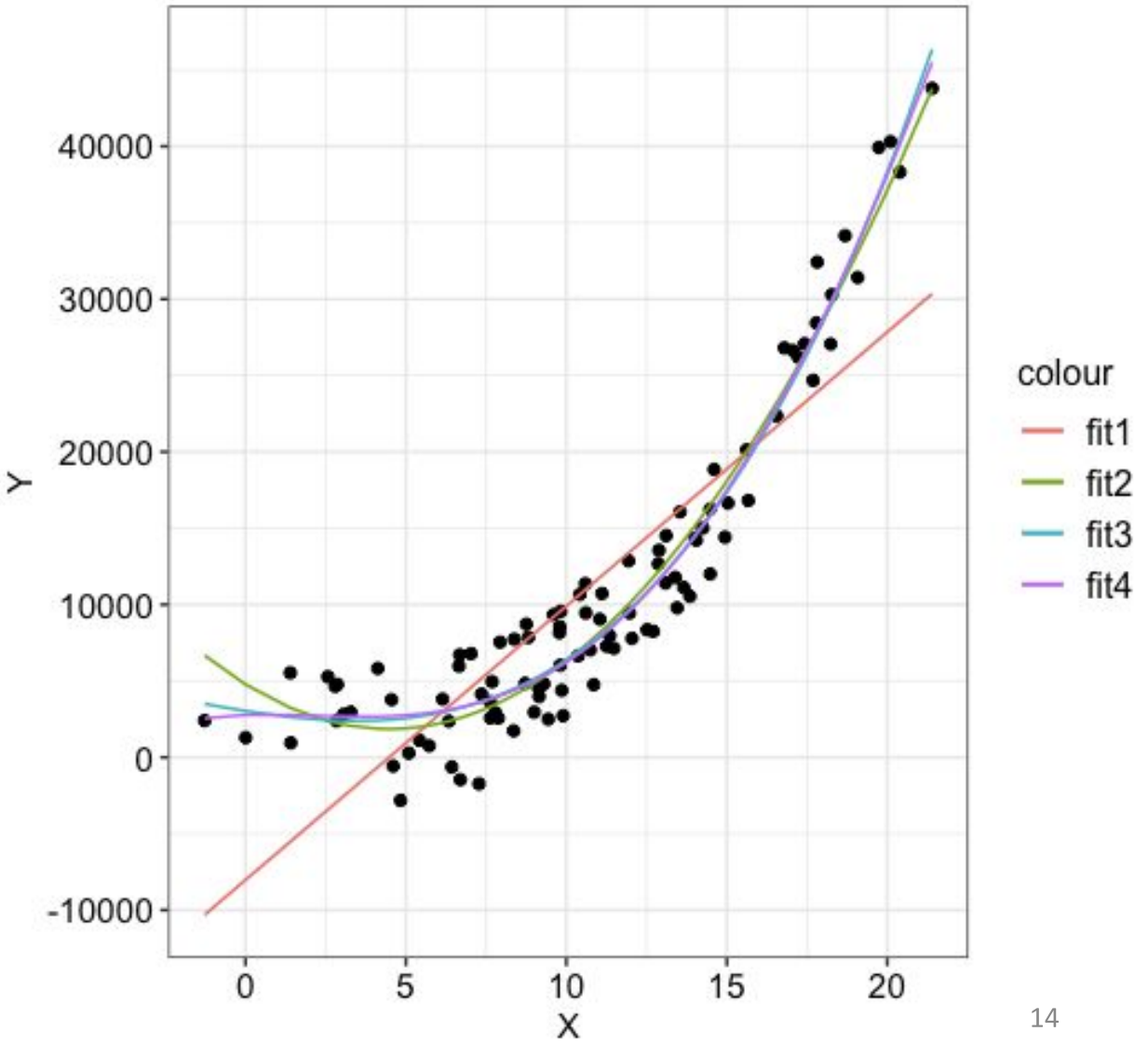
However,

1) Good quality data (and features) are more important
2) Adhering to the basic principles comes first
3) After that, certain algorithms tend to work somewhat better than others

Back to overfitting versus underfitting

Suppose we try to find out the model with the best fit for a given sample

We would somehow have to choose between different models

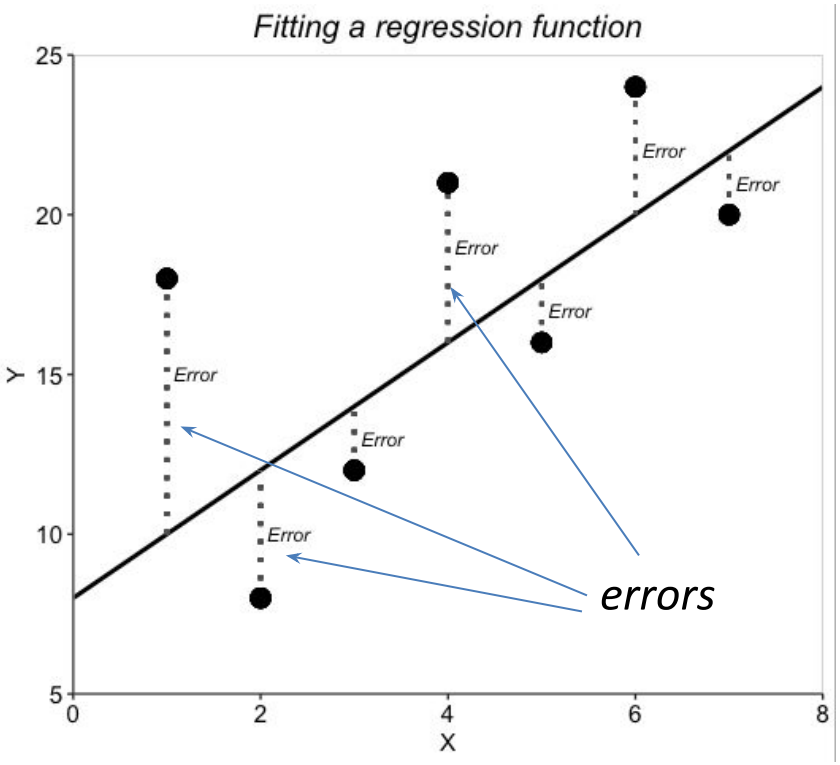


How to determine the best model?

The lower the mean squared error, MSE, the better the model fits the data

The MSE is literally computed as the mean of the squared errors

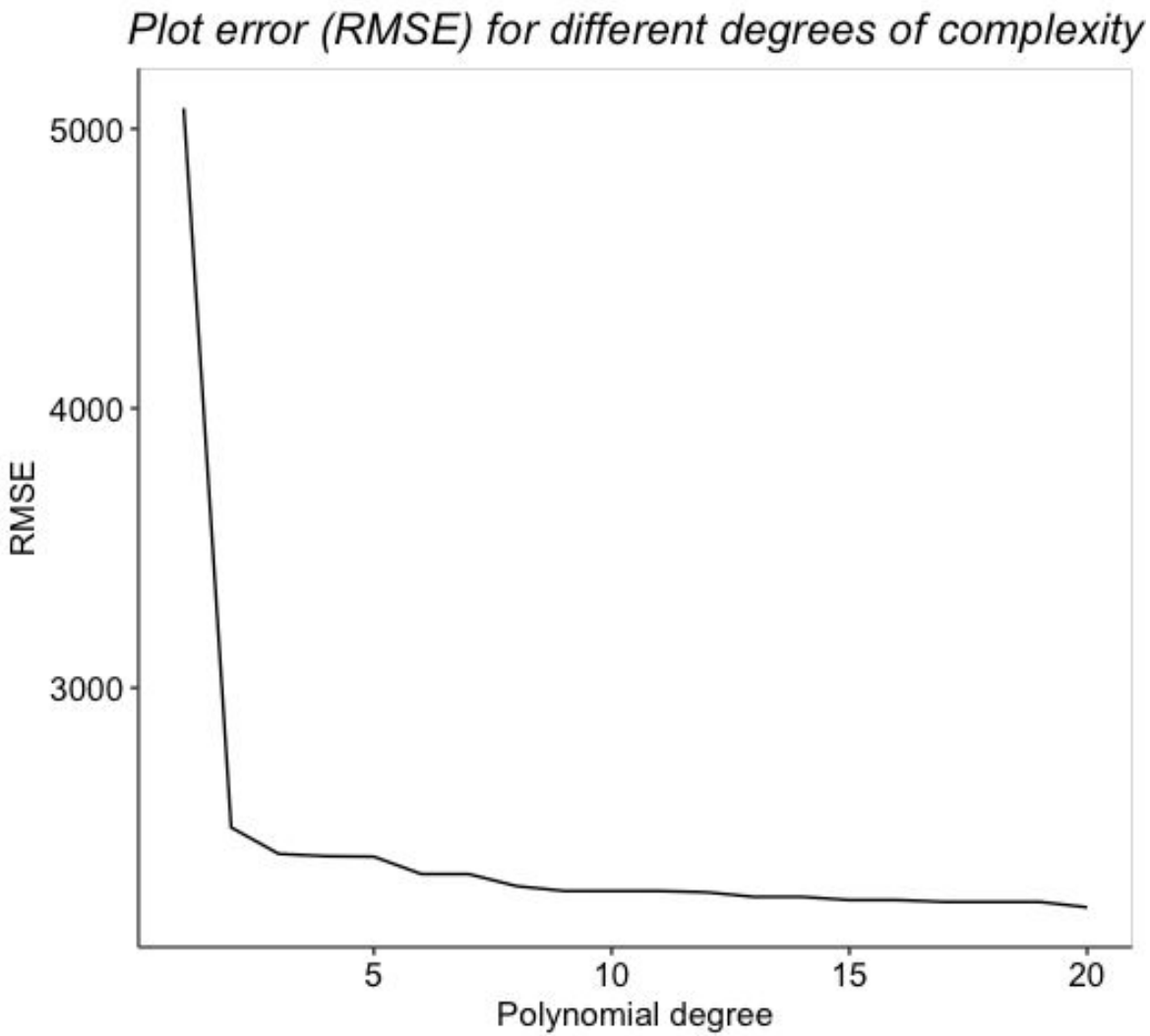
Similarly, the root of the MSE (RMSE) is a metric for which a lower value corresponds to a better model fit



How to determine the best model?

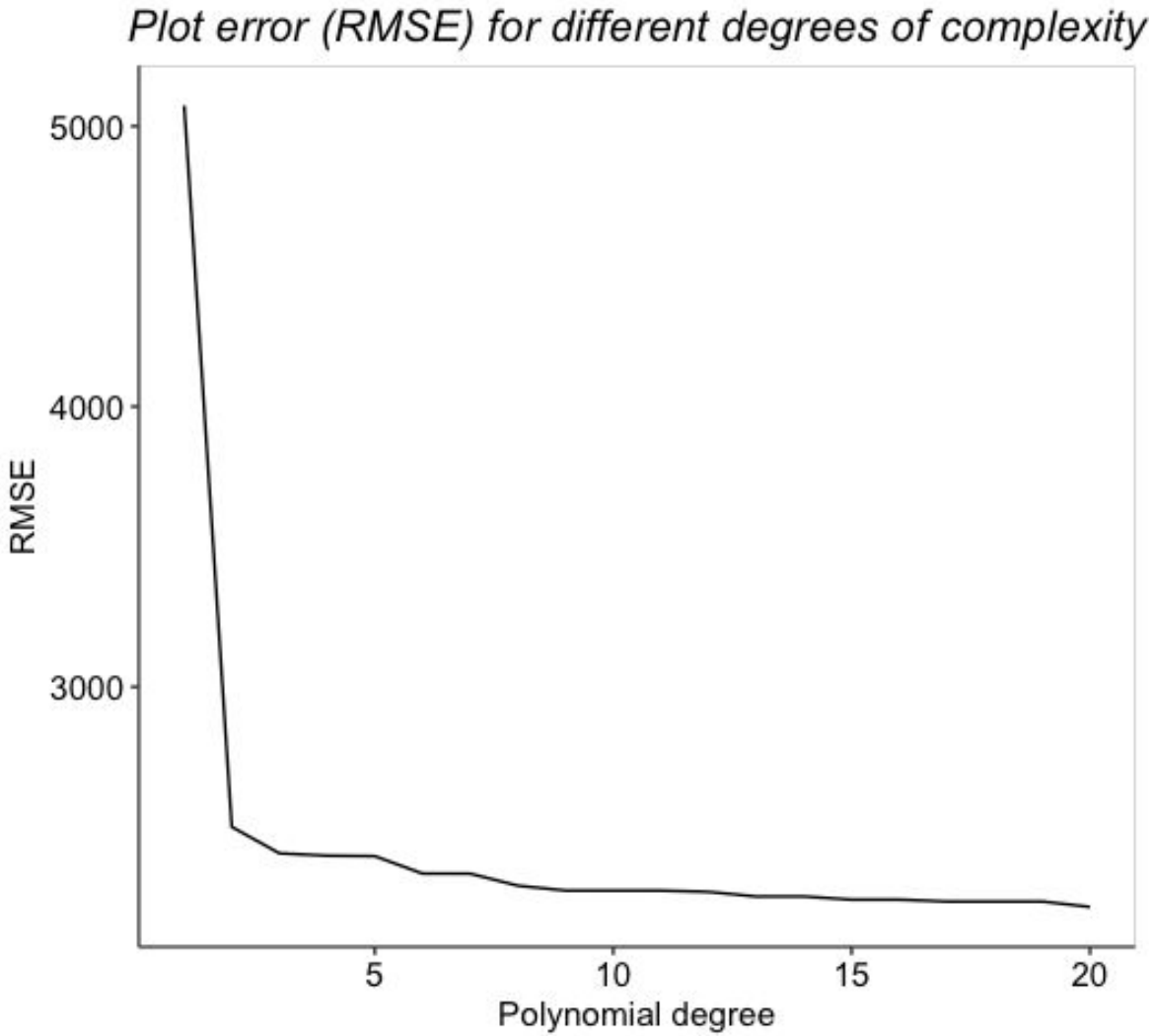
Suppose we keep trying more and more complex models, simply by adding polynomials (X-axis) and look at what happens to the fit (i.e. RMSE)

We would somehow have to choose between different models



How to determine the best model?

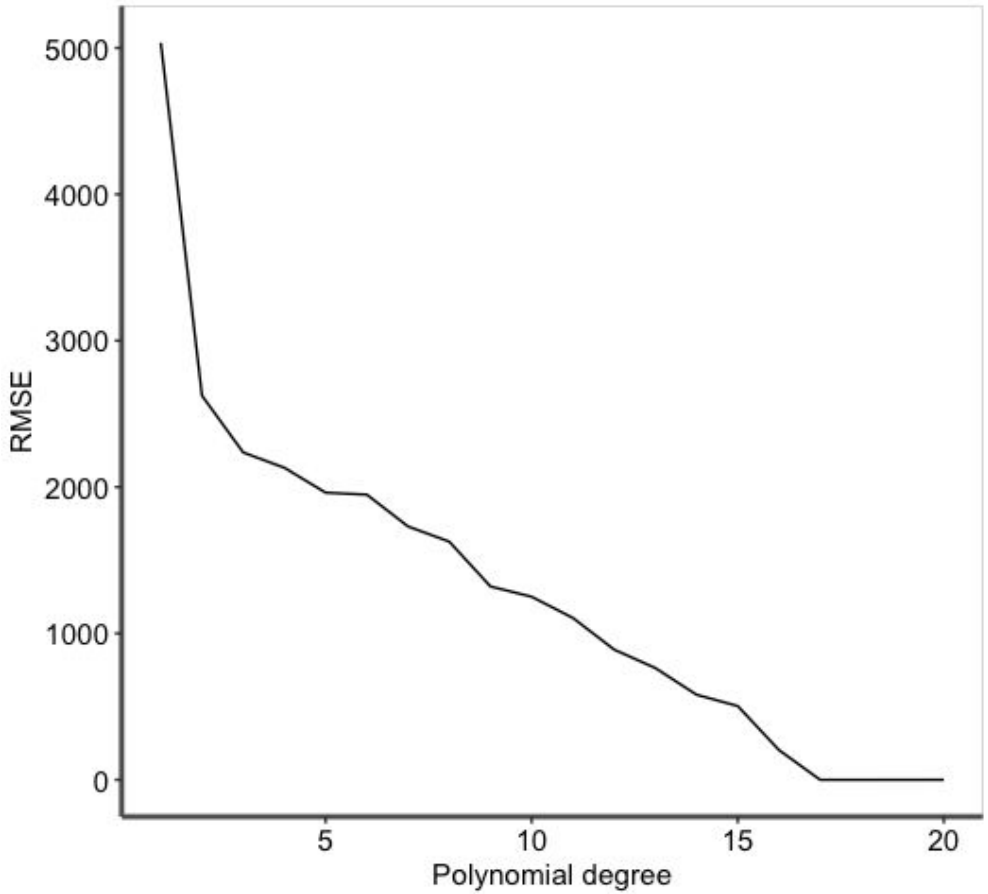
1. What can you conclude from the results?
2. What can you <u>not</u> conclude from the results?



How to determine the best model?

Suppose we add random variables (unrelated to Y) and at them as well with higher and higher polynomial degrees

1. What kind of risk becomes apparent in this output?



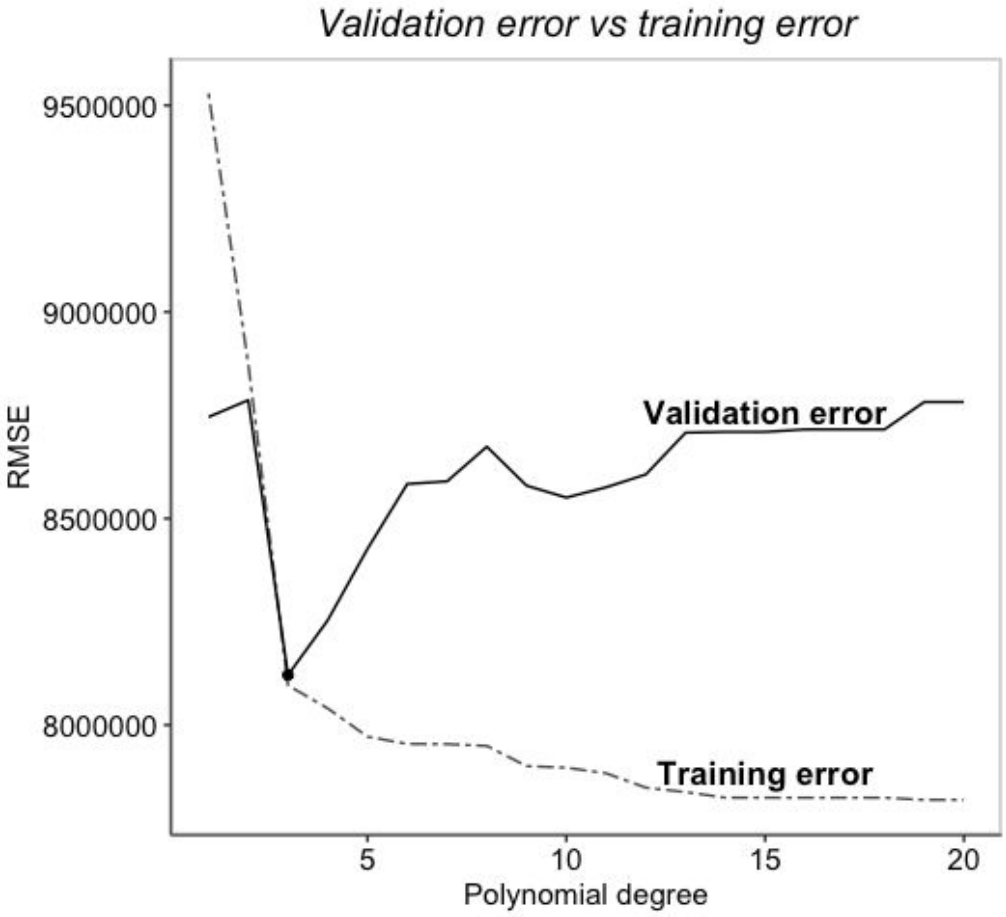
How to determine the best model?

Suppose we split the data in a training and validation set, train models with higher and higher polynomial degrees, and test their fit on the validation set

Suppose we know that the 'true model' is defined with a polynomial of degree 3, as:

$$Y = 250 + 20X - 2500X^2 + 15X^3$$

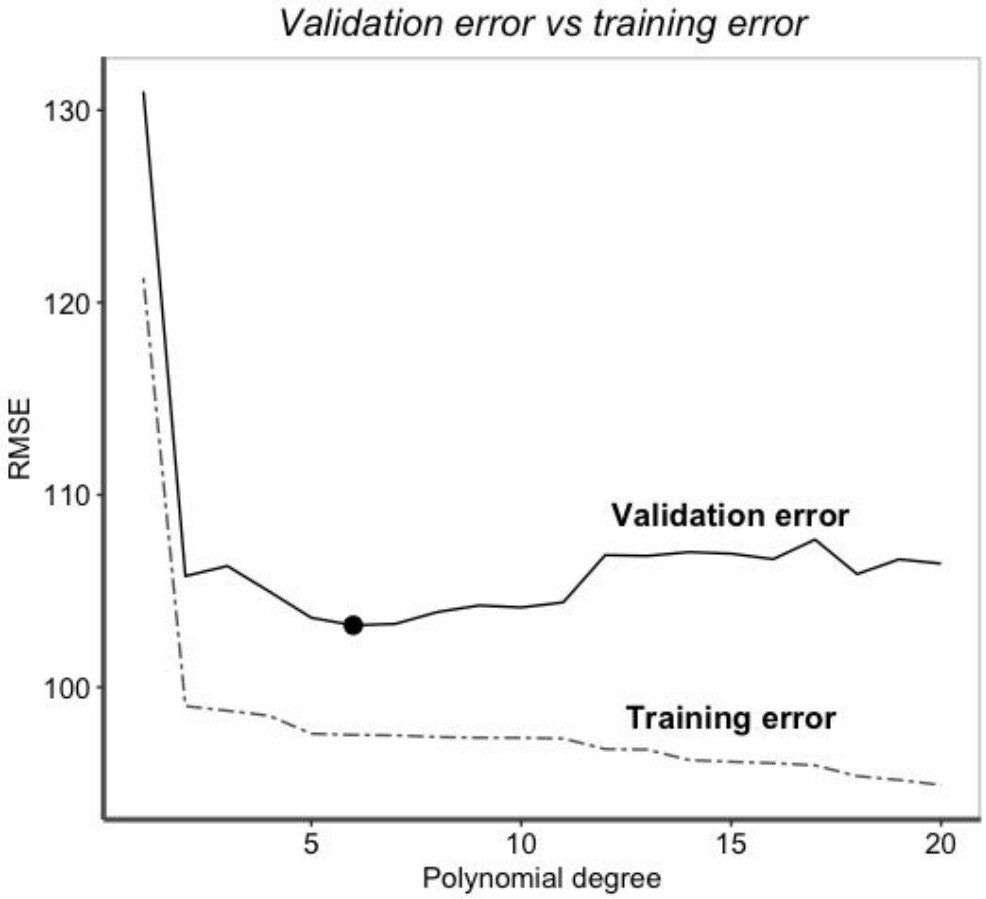
1. What do we learn from this output?



How to determine the best model?

Suppose we have the same ‘true model’, but run our analysis again on a different dataset.

1. What do we learn from this output?



overfitting.R – lessons learned

Section 1: adding more input variables always leads to a better model fit (on the data on which the model was trained)

Section 2: this will eventually lead to a perfect fit (overfitting), even when the input variables are in reality unrelated to the outcome Y

Section 3: A validation set helps to guard against overfitting

Section 4: A validation set helps to guard against, but is no guarantee against, overfitting

Overfitting – summary (1/2)

1. Overfitting is about modelling noise, on top of the signal in the data
2. Overfitting occurs when a model is too flexible (compared to the problem at hand)
3. Overfitting leads to a model that fits well on the training data, but poorly on new (test) data
4. Overfitting leads to weak generalizability, thus compromising the one thing we strive for

Overfitting – summary (2/2)

For every machine learning application the data scientist should search for the optimal level of flexibility / complexity of the machine learning model, to ensure best possible optimal generalizability

Practically speaking, simply consider a wide enough range of flexibility / complexity to be sure it contains the optimal level of flexibility / complexity

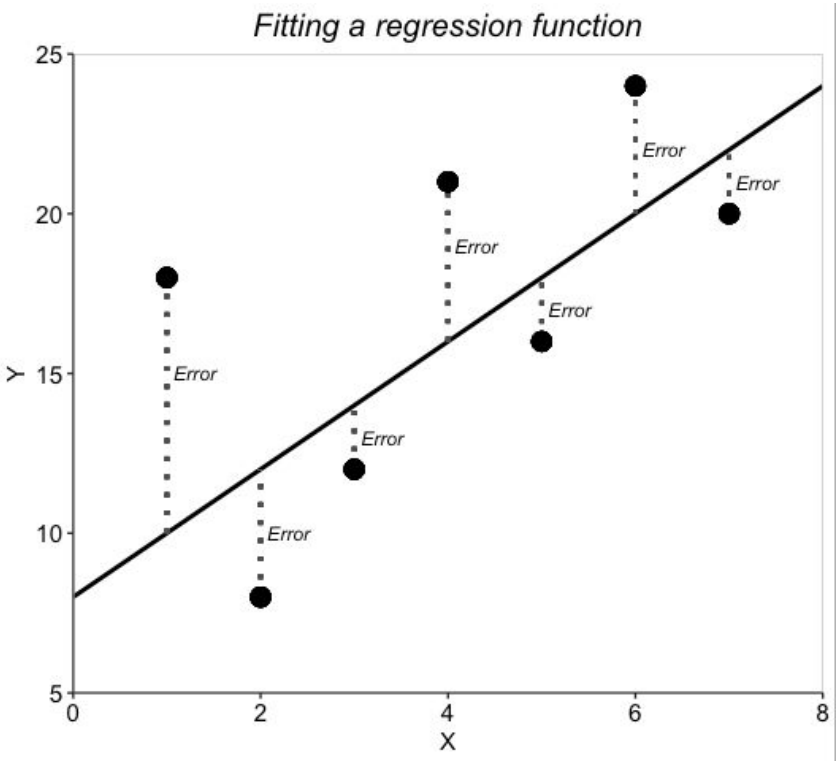
The optimal level of flexibility / complexity is where the performance on the validation set is optimal. Lower levels of flexibility correspond to underfitting; higher levels correspond to overfitting

Test MSE & Bias-Variance Tradeoff

Test MSE (Mean Squared Error)

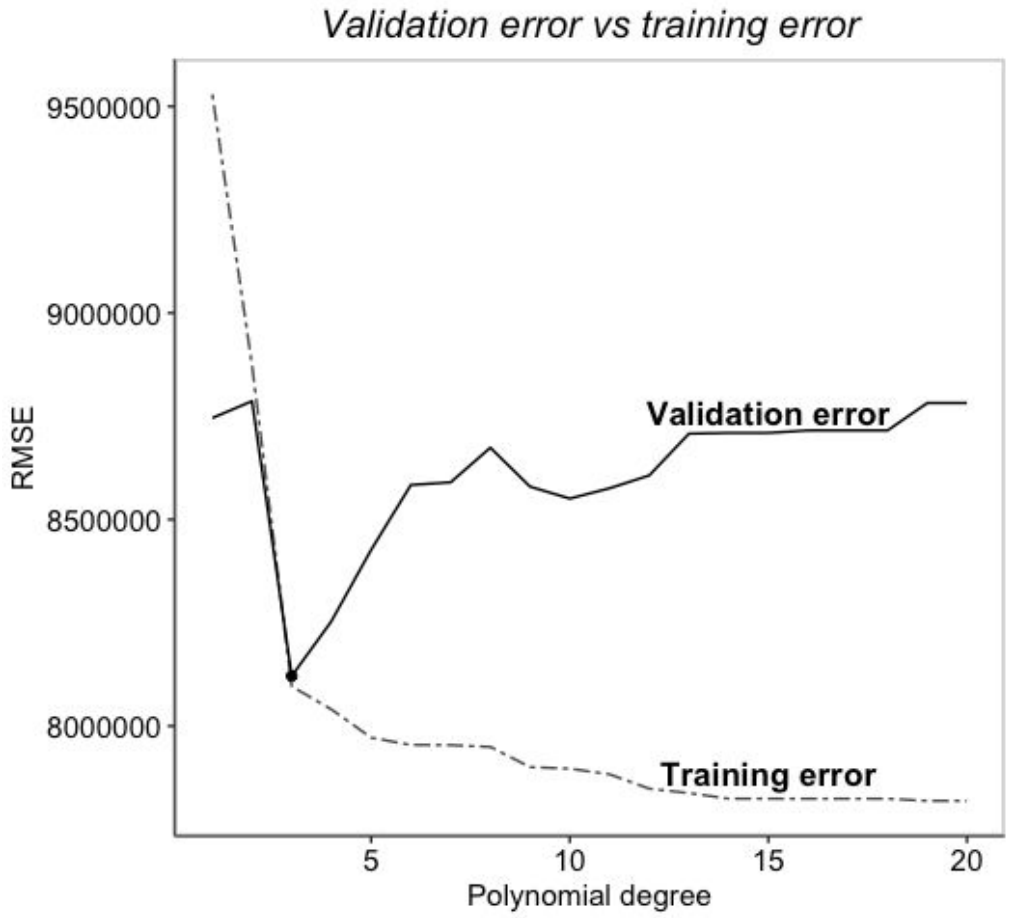
Test MSE: a measure for how much the model deviates from the data for a new (test) dataset, which was not used to train the model

The lower the test MSE, the better the model fits new data



Looking deeper into the Test MSE

What is happening here exactly?

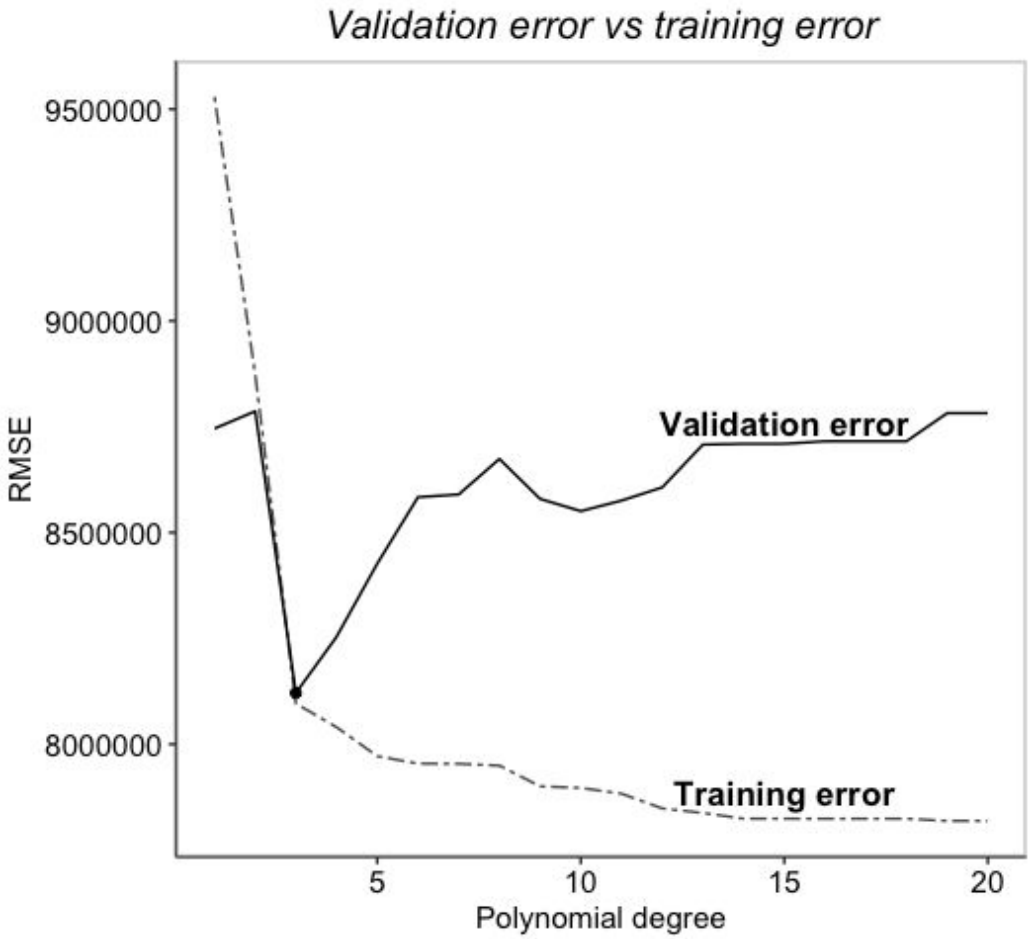


nstitute

Looking deeper into the Test MSE

The test MSE improves as long as the increase in flexibility better enables the model to capture the structure of the data (the signal),

while the test MSE gets worse when increasing flexibility does not lead to a better approximation of the structure of the data



Looking deeper into the Test MSE

Bias – Variance Decomposition

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon). \qquad (2.7)$$

Taken from ISLR

Expected Test MSE = Variance of the statistical learning method + Bias^2 of the statistical learning method + Irreducible error

Bias – Variance Decomposition

Expected Test MSE = Variance of the statistical learning method + Bias^2 of the statistical learning method + Irreducible error

Expected Test MSE: how well do we expect the trained model to fit new data?

Variance of the statistical learning method: degree to which the model changes when it is trained on a different sample of training data

Bias of the statistical learning method: degree to which the model *is not able* (is not flexible enough) to capture the true relationship between the X's and Y ⇔ degree to which the model oversimplifies reality

Irreducible error: variation in the outcome Y that cannot be explained by the X's ⇔ the lower bound of the test MSE

Bias – Variance Decomposition

Expected Test MSE = Variance of the statistical learning method + Bias^2 of the statistical learning method + Irreducible error

Flexibility is key: increasing flexibility will decrease bias and increase variance

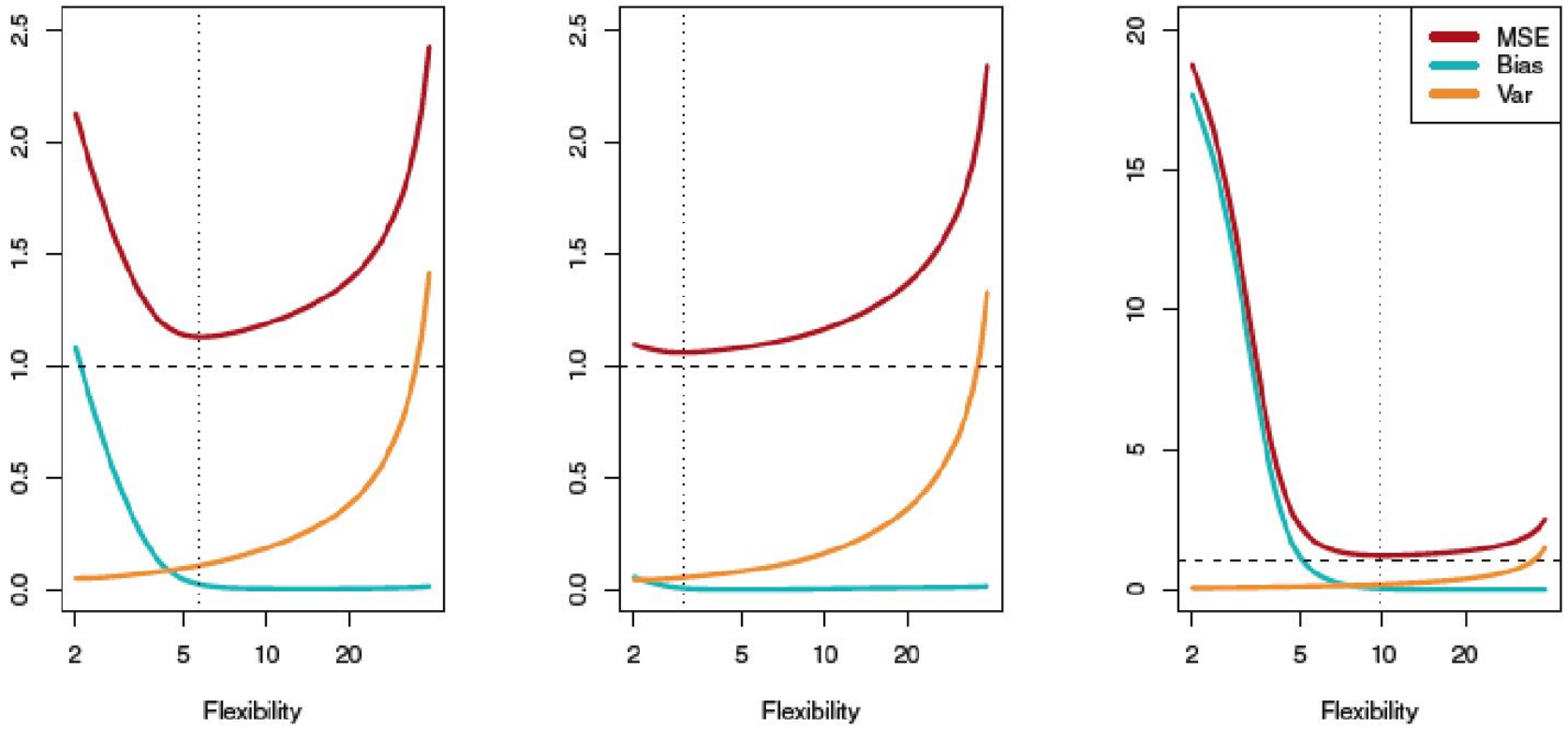
Best performing model (lowest test MSE) manages to find the sweet spot between low bias, and not too high variance

The optimal level of flexibility could be low, high and anything in between, and depends on the underlying structure

Estimating machine learning models is therefore about starting with low flexibility, and slowly increasing flexilibility up to the point where the reduction in bias^2 is stronger than the increase in variance

BIAS – VARIANCE TRADEOFF

Bias – Variance Tradeoff – examples



Taken from ISLR, figure 2.12

Q1: What is the optimal level of flexibility in each figure?

Q2: What can we say about the true relation between input and output in each figure?

Conclusion

The aim of supervised machine learning is to develop a model that:

- generalizes well to new data
- makes good predictions / classifications on new data
- has the lowest possible test MSE

The test MSE is made up of both the bias and the variance of the model, so estimating a machine learning model is always about finding the optimal balance between bias and variance

The more flexible the algorithm, the lower the bias
The more flexible the algorithm, the higher the variance

Optimizing a supervised machine learning model therefore comes down to considering different degrees of flexibility

Final thoughts on the bias-variance trade-off

The bias-variance trade-off helps to understand the performance of your models better:

- “multicollinearity in your data is undesirable, as it increases the variance of your model, without reducing bias”
- “Random forest works very well as compared to a single decision tree, as it greatly reduces variance, while only slightly increasing bias”

Practically, a relatively inflexible model is often used to quickly constitute a reference model, against which various other models are compared, where for such models much effort is put into finding the optimal balance between bias and variance.

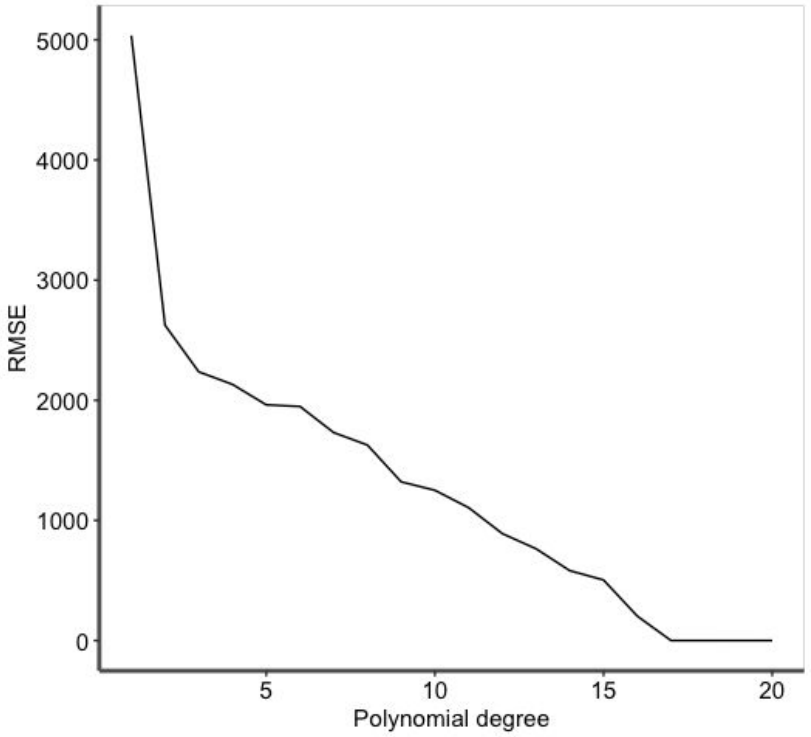
Splitting the Data

Overfitting leads to two errors

1. The wrong model is chosen
2. The performance is overestimated

Applying machine learning is about setting up a process for proper:

- *model selection*, and
- *model assessment*



The importance of a good strategy

The importance of a good strategy around model selection and model assessment can not be overstated:

Remember:

- Machine learning is a process of trial-and-error
- This can only work with the availability of honest feedback during this process of trial-and-error
- Honest feedback of a model's performance can be attained by carefully validating the impact of different modelling choices, through splitting the data

Work out a specific strategy for this part of your analysis!

Splitting the data

Honest evaluation of the performance of your model (model assessment) is achieved by splitting the dataset into a training set and test set

A rule of thumb is to use 70%-80% as training data and 20-30% as test data

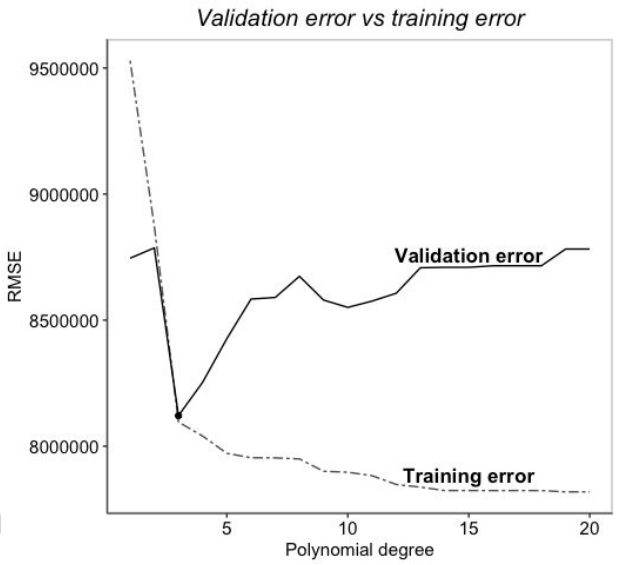
Splitting the data

A test set will *only* give a good estimate of the performance of the model, if the development of the model did not use data from the test set

TRAINING SET	TEST SET

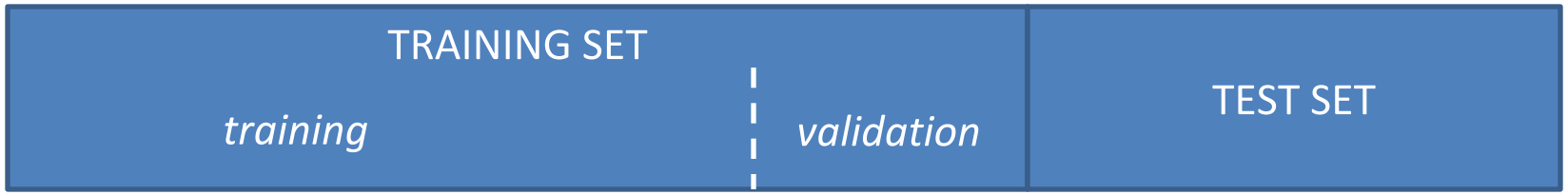
See how this was not the case in the examples so far

Q: why is the optimal MSE found in the figure probably not a good estimate of the test MSE?



Splitting the data

Therefore the TRAINING SET itself is being split into a *training* and *validation* set:



Training and *validation* are used to determine optimal model settings (**model selection**) in a process called tuning the hyperparameters

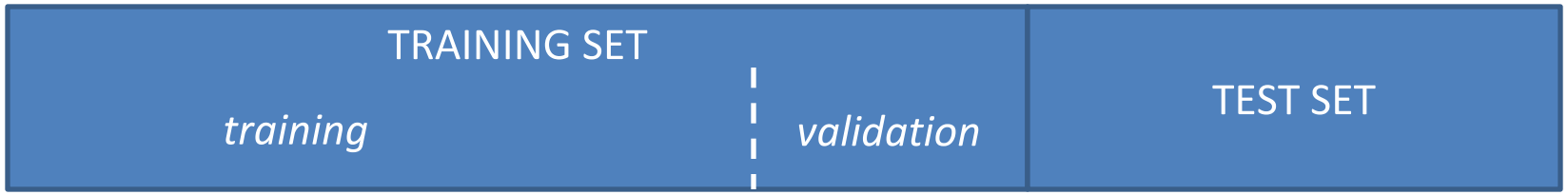
Next, the entire TRAINING SET is used to re-estimate the model with the optimal parameter settings

Applying this model to the TEST SET gives a reliable estimate of the model's performance on new data (**model assessment**)

Performance on the test set is expected to be somewhat worse than the performance on the TRAINING SET

Splitting the data

Therefore the TRAINING SET itself is being split into a *training* and *validation* set:



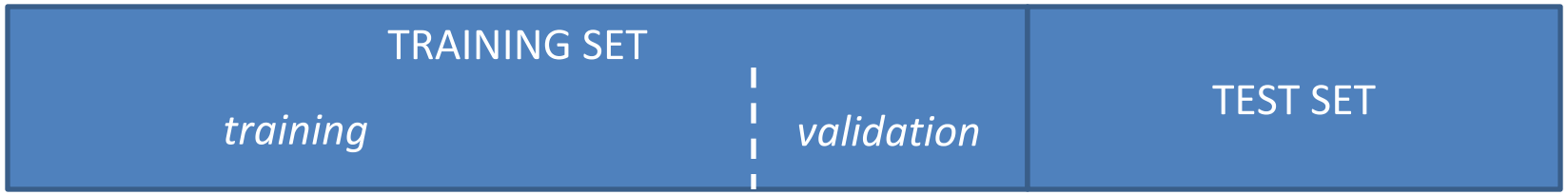
Training and *validation* are used to determine the optimal model settings (**model selection**)

This means that practically every modelling choice you face can be tested in this phase

Splitting the data

Splitting the data into the TRAINING SET and TEST SET needs to be done 'manually'

Splitting the TRAINING SET into *training* and *validation* happens 'automatically'



The split between TRAINING SET and TEST SET can be done randomly, while ensuring that the outcome Y is evenly distributed over both sets

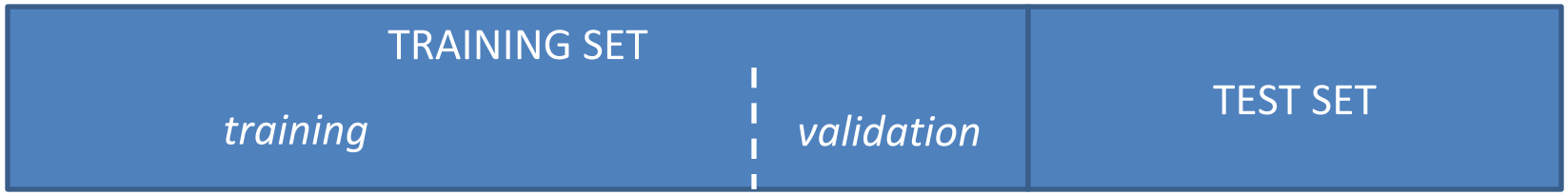
In case of a clear time component in your data, you could use data in later years as TEST SET

In case of a clear location component in your data, separate locations could be used as TEST SET

Splitting the data

Splitting the data into the TRAINING SET and TEST SET needs to be done 'manually'

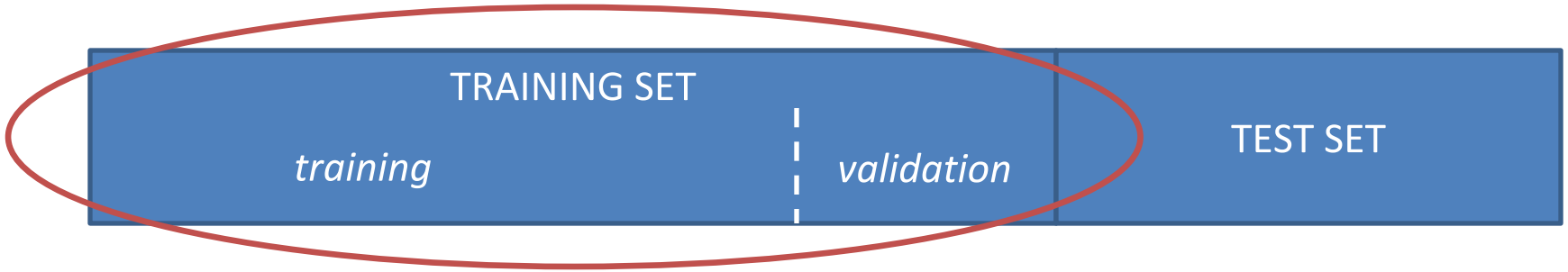
Splitting the TRAINING SET into *training* and *validation* happens 'automatically'



Be careful to construct the TEST SET in such a way that prediction/classification in this set does not become easier than it would be in a real-life setting

Splitting the TRAINING SET data

Splitting the TRAINING SET into *training* and *validation* can be done in a number of ways

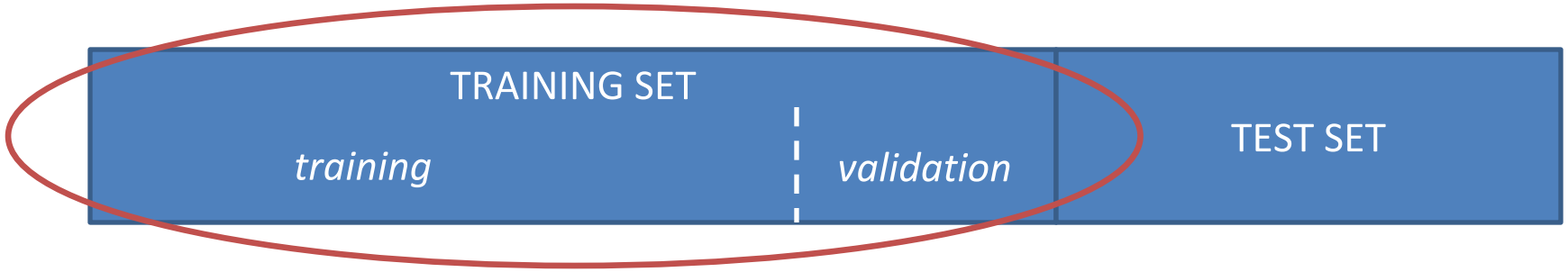


The consensus is to use 5 or 10-fold cross-validation

If that is problematic, the validation-set approach could be taken

Validation Strategies

Splitting the data

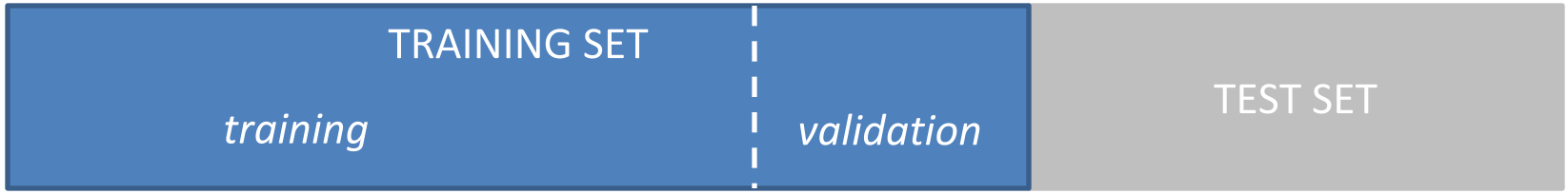


To recap: validating your model, which is trained on the *training* set, is done to:

- determine the optimal level of flexibility (by tuning the hyperparameters)
- test any modelling choice you face

⇒ This is all part of the **model selection** phase.

1: Validation Set Approach



The Validation Set Approach splits the data in the TRAINING SET into two parts: data for training and data for validating the model (e.g. 70% versus 30%)

This method is straightforward, but has two drawbacks:

- The model does not make optimal use of all the data available in the TRAINING SET
- The outcomes can depend strongly on the specific split between *training* and *validation*

2: Leave One Out Cross Validation (LOOCV)

LOOCV goes through multiple iterations of training the model and validating, each time leaving one different observation out for validation and using all other observations for training the model.

- All data in the TRAINING SET is used for estimating the model
- There is no randomness in what data is used for training and validation (exhaustive method)

LOOCV is an option when there is very little data

3: K-fold Cross Validation

K-fold Cross Validation is a good compromise between the previous two strategies

K-fold cross-validation using k=5 or k=10 is seen as a best practice

As k-fold cross validation is non-exhaustive (different splits could lead to different results), this process is often repeated several times (repeated cross validation)

To Conclude

To conclude

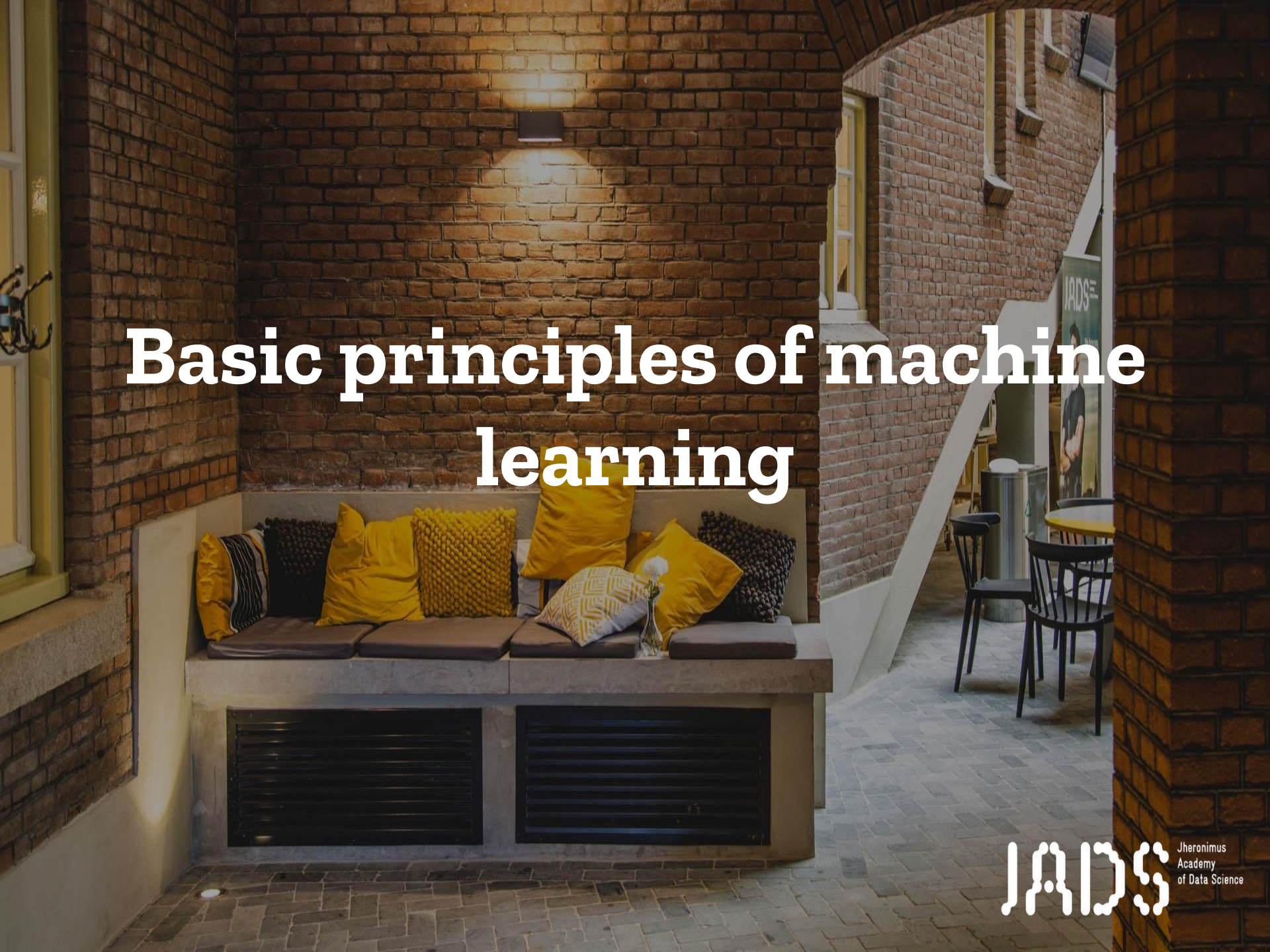
A supervised machine learning model aims for one thing: to work well on new data

This should be achieved by fully separating the phase of training the model from the judging of its performance (separating **model selection** and **model assessment**)

Model selection is done by trying out different models and searching for the optimal level of flexibility within the TRAININGSET

This requires the use of validation data, with 5 or 10-fold cross validation being the most applied strategy

=> These basic principles apply just as much to regression as to classification problems


Basic principles of machine learning
JADS
Jheronimus Academy of Data Science