Machine Learning:: Model Evaluation / Bias Variance Tradeoff

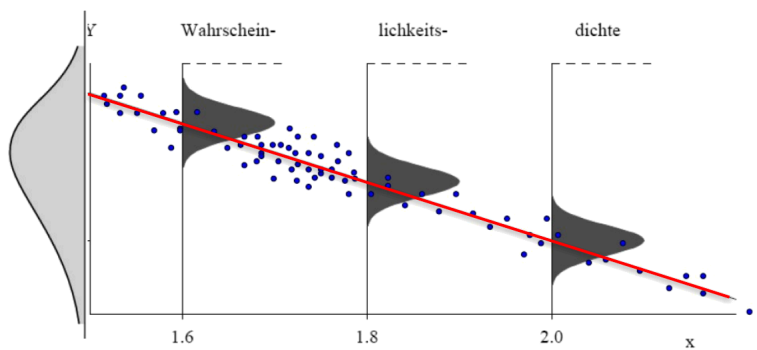
Dozenten: Prof. Dr. M. O. Franz, Prof. Dr. O. Dürr

Remark: Much of the material have been developed together with Beate Sick (ZHAW/Uni Zurich).

Evaluating Predictions

Regression

1. Deterministic: Assigns X to a Number
   - Examples: Linear Regression (when done badly)
   - Performance Measure: **Mean Squared Error, Mean Absolute Error**
2. Probabilistic Regression: Assigns X to a probability
   - Examples: Linear Regression
   - Performance Measure: **NLL on testset** (or other proper scoring rules see DL)



Classification: three kinds of output

Deterministic: X → a class — e.g. perceptron, decision tree — measure: accuracy, confusion matrix

Score based: X → a score per class — e.g. SVM (to come) — measure: ROC curve, AUC
higher = "surer"

Probabilistic: X → a probability per class — e.g. logistic regression, (naive) Bayes — measure: NLL on the test set (proper scoring rule)

Probabilistic ⊂ Score based ⊂ Deterministic: a probability is also a score, and a threshold on a score gives a class. So an inner model can be evaluated with all outer measures, but not the other way round.

Deterministic: confusion matrix and derived values

	predicted positive	predicted negative
actual positive	TP	FN
actual negative	FP	TN

- Accuracy $= (TP + TN) \,/\, N$
- Precision $= TP \,/\, (TP + FP)$: how many alarms are real?
- Recall = sensitivity = TPR $= TP \,/\, (TP + FN)$: how many positives are found?
- Specificity $= TN \,/\, (TN + FP)$
- FPR $= FP \,/\, (TN + FP) = 1 -$ specificity
- F1 $= 2 \cdot \text{precision} \cdot \text{recall} \,/\, (\text{precision} + \text{recall})$
- … many more

Example, Los Angeles: 360 days of sun and 5 days of rain a year. Can you build a classifier for tomorrow's weather with accuracy > 98 %?

Classifier "always sun":

	pred. rain	pred. sun
rain	0	5
sun	0	360

- Accuracy $= 360/365 = 98.6\,\%$ 🎉
- Recall $= 0/5 = 0$: it never finds rain
- Precision $= 0/0$: undefined, F1 $= 0$

With imbalanced classes, accuracy alone says little.

Score based: AUC as a ranking probability

We have a score based classifier, the higher the score, the more the model "thinks" the example belongs to class 1.

Random pair: s_+ = score of an example from class 1, s_- from class 2. Ties count half:

$$\mathrm{AUC} = P(s_+ > s_-) + \tfrac{1}{2} P(s_+ = s_-)$$



4 × 8 = 32 pairs

green + ½ gray = 28

AUC = 28/32 = 0.875

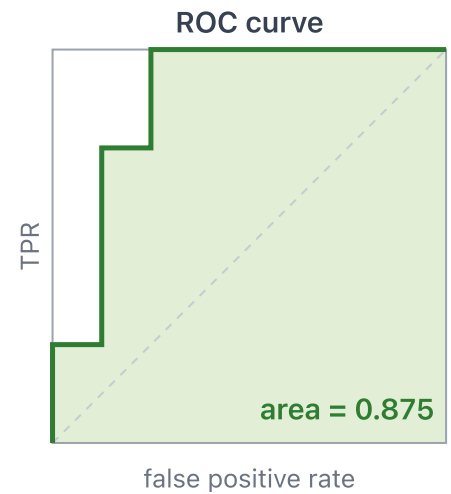
class 1 higher: 1

tie: ½

class 2 higher: 0

LA: always sun

reset



Once class 1 is above all of class 2, the size of the gap no longer changes the AUC.

Probabilistic: the NLL (reminder)

The classifier gives a probability $\hat{p}(y_i \mid x_i)$ for the true class. On the **test set**:

$$\text{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log \hat{p}(y_i \mid x_i)$$

Loss of one example:

$\hat{p}$(true class)	0.9	0.5	0.01	0
$-\log \hat{p}$	0.11	0.69	4.61	∞

Confident and wrong is expensive.

Los Angeles again: 5 days of rain a year. What is a naive probabilistic classifier, and what is its NLL?

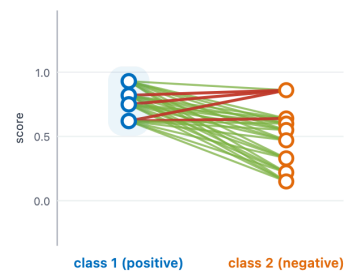
- "always sun" as a probability: $\hat{p}(\text{rain}) = 0$
  NLL $= \infty$ (one rainy day is enough)
- $\hat{p}(\text{rain}) = 5/365$ every day: NLL $= 0.072$
- A real weather model must beat 0.072

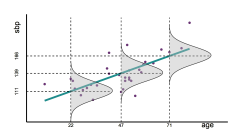
> The NLL (log score) is a **strictly proper scoring rule**: its expected value is best only if the model reports the true probabilities. Accuracy and AUC are not proper (AUC only sees the ranking, not the calibration). More on proper scoring rules in the DL part.

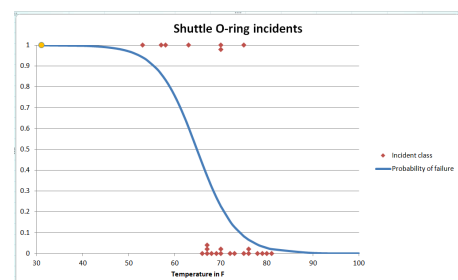
Summary: Performance Measures for Classifiers

- Deterministic Classifier (Perceptron)
  - Confusion Matrix Based
    - Accuracy
    - Precision, Recall, F1 Score, …
- Score based classifiers (SVM)
  - ROC
    - AUC and various others
- Probabilistic classifiers (Lin/Log. Regression)
  - more on this in DL
  - Likelihood
    - Negative Log Likelihood (per example)
  - Likelihood is not limited to classification

		True class: Positive	True class: Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN







PAUSE BIS 12:20

Crossvalidation



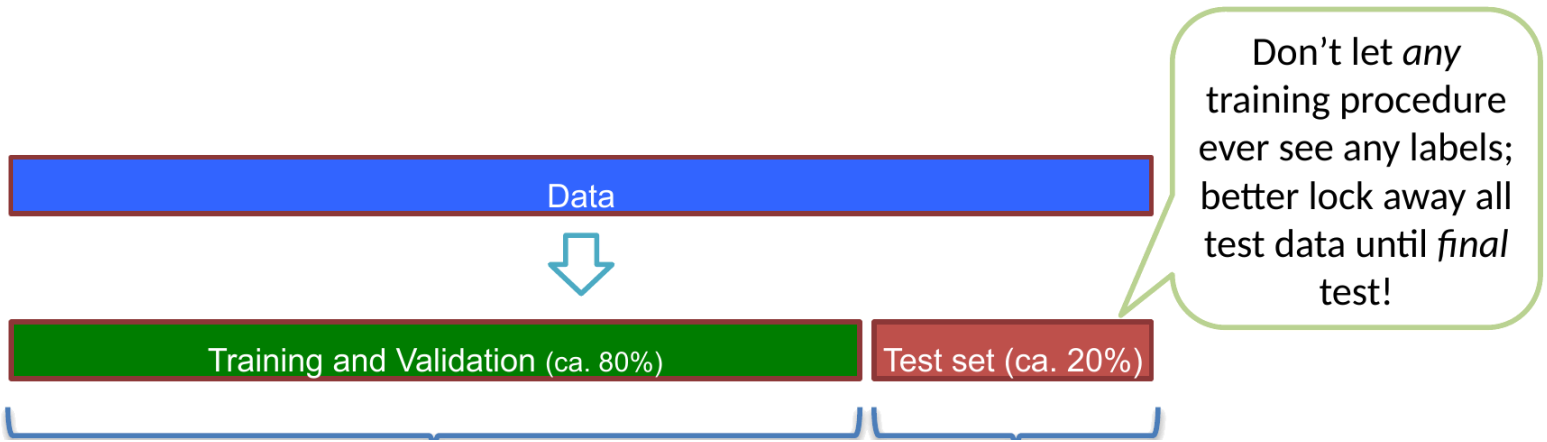
Questions for X-Validation

- Model Selection (which model to take)
  - Often there is a knob controlling the complexity of a classifier
    - Number of NN in KNN?
  - Number of features to use?
  - Square / log the features and add them?
  - Shall I use Logistic Regression or Perceptron?
- Model Evaluation
  - How good is the performance on new unseen data.
    - Note that the performance can also be
      - AUC instead of Accuracy
      - Likelihood of observations instead of AUC or accuracy



How to be on the safe side

Typical strategy, spare some data for the final testing.

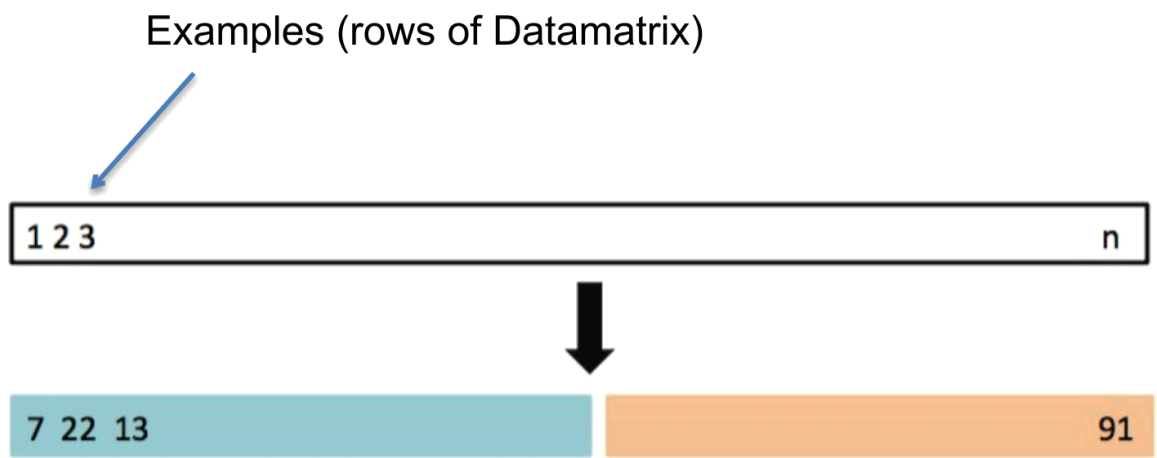


Use this to train model, select models and hyperparameters

Use this to estimate the performance.

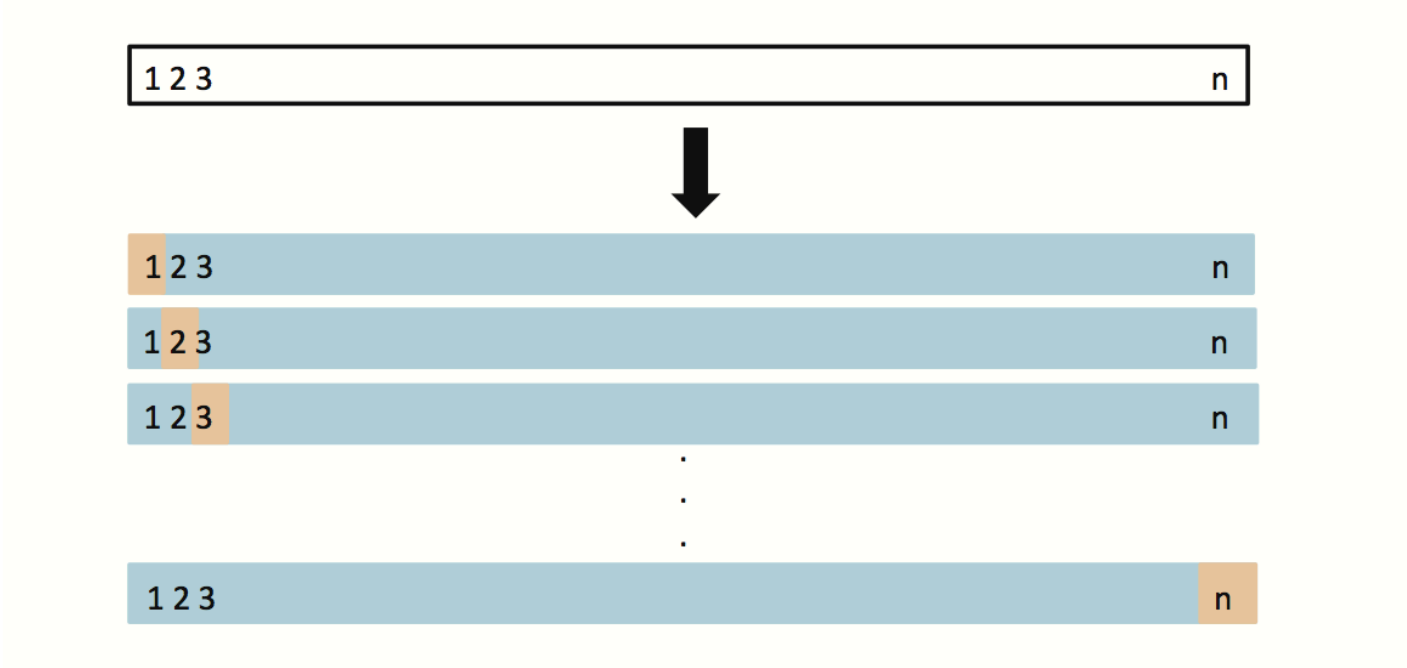
In the following, we are only talking about Training and Validation set.

The Validation Set Approach



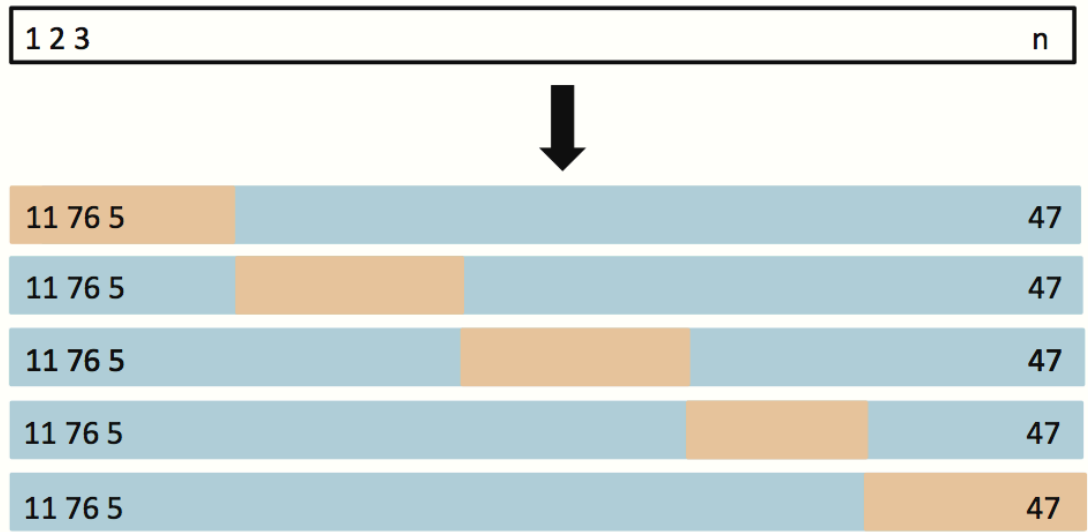
A random splitting into two halves: left part is training set, right part is validation set

Leave-One-Out Cross Validation (LOOCV)



Fit w/o red sample and predict the red sample. Average over all n repeats

K-fold Cross Validation

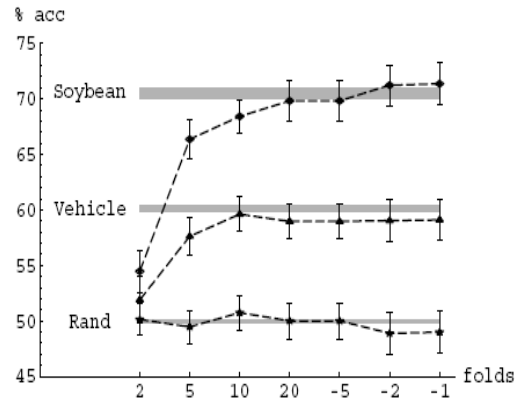


Fit w/o red samples and predict the red samples. Average over all k repeats. Do a weighted average if folds do not have the same size.

Question: What happens if k=n, what if k=2?

What is the best cross-validation scheme?

- LOOCV
  - Not random
  - Sometimes slow (there are build in procedures for some classifiers)
  - Higher Variance (between completely new data sets)
  - Lowest possible bias (nearly the whole data seen)
- K-Fold Xvalidation
  - Random result
  - Usually faster
  - Lower Variance
  - Higher Bias (usually not problematic)



Standard today: k=10 Fold cross-validation (sometimes repeated after permutation)

Practical Tipps: Table 1 with Error Bars and Null model

Table 1: Performance comparisons across different methods (columns) and datasets (rows) for the LPPD and RMSE metric. The method closest to the MCMC gold standard is highlighted in **bold**.

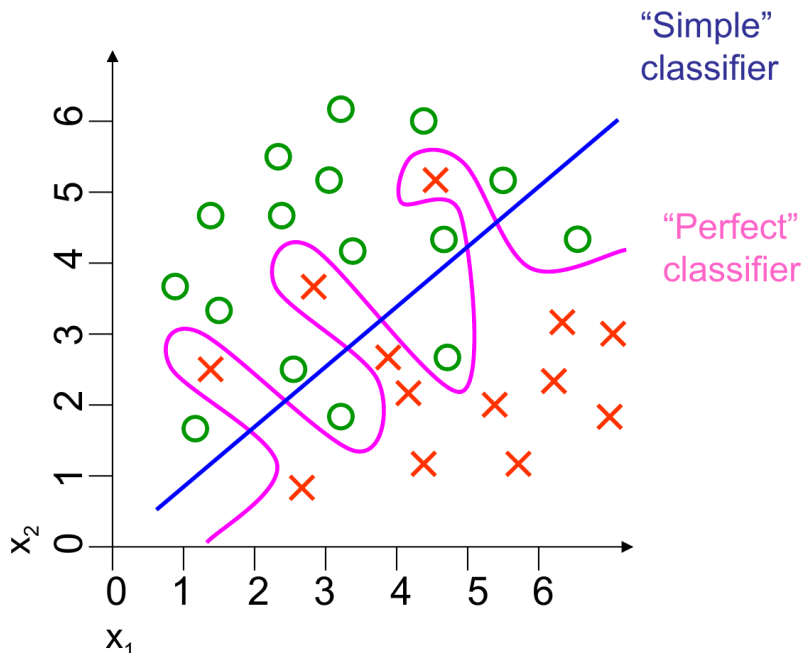
		MCMC	DE	LA	ModeCon	Tunnel-2	Tunnel-20
LPPD (↑)	airfoil	0.67 ± 0.04	-1.09 ± 0.05	-3.05 ± 0.16	0.09 ± 0.05	0.25 ± 0.09	**0.34** ± 0.04
	bikesharing	0.28 ± 0.04	-0.94 ± 0.03	-2.83 ± 0.11	0.02 ± 0.03	0.12 ± 0.04	**0.14** ± 0.03
	concrete	0.09 ± 0.18	-0.94 ± 0.07	-2.99 ± 0.27	-0.29 ± 0.14	-0.08 ± 0.05	**-0.04** ± 0.09
	energy	2.18 ± 0.07	-0.91 ± 0.07	-2.90 ± 0.19	1.59 ± 0.05	**1.74** ± 0.14	1.70 ± 0.28
	yacht	2.90 ± 0.19	-0.91 ± 0.14	-2.91 ± 0.63	**1.33** ± 0.80	0.75 ± 3.01	**1.33** ± 1.75
RMSE (↓)	airfoil	0.15 ± 0.02	0.30 ± 0.02	0.27 ± 0.03	0.22 ± 0.01	0.19 ± 0.01	**0.18** ± 0.01
	bikesharing	0.21 ± 0.01	0.26 ± 0.01	0.25 ± 0.01	0.25 ± 0.01	**0.23** ± 0.01	**0.23** ± 0.01
	concrete	0.25 ± 0.02	0.31 ± 0.01	0.44 ± 0.03	0.31 ± 0.02	0.30 ± 0.02	**0.28** ± 0.02
	energy	0.03 ± 0.00	0.07 ± 0.01	0.06 ± 0.01	0.05 ± 0.00	**0.04** ± 0.01	0.05 ± 0.01
	yacht	0.04 ± 0.01	0.13 ± 0.03	0.13 ± 0.04	0.06 ± 0.01	0.05 ± 0.01	**0.04** ± 0.01

Always include a null model (a baseline without features) as a column. Los Angeles: "always sun" (accuracy 98.6 %), constant $\hat{p}(\text{rain}) = 5/365$ (NLL 0.072). A model is only worth something if it beats the null model.

Bias Variance Tradeoff

“Perfect” Vs. “Simple” classifier





Which is better?

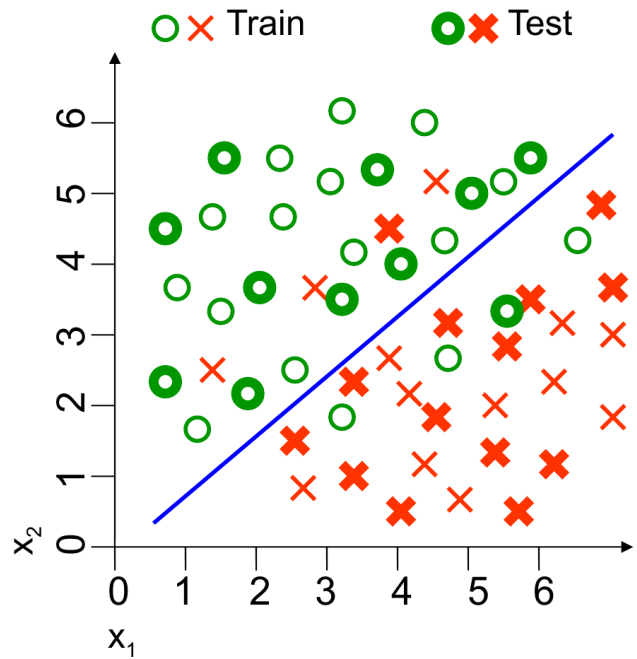
Check on a test-set (don’t use all your labeled data to train)

Cross validation of the “simple” classifier



Training set:
6/29=20%
misclassification

Test set:
2/25=8%
misclassification

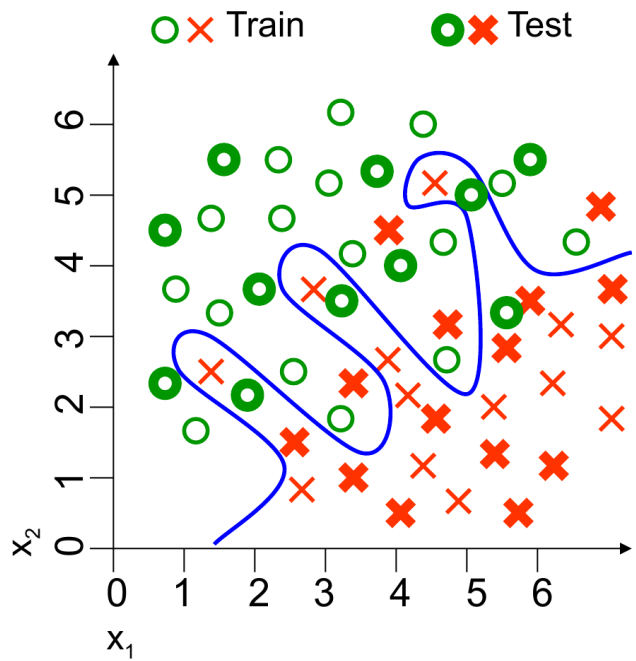


Cross validation of the “Perfect” classifier

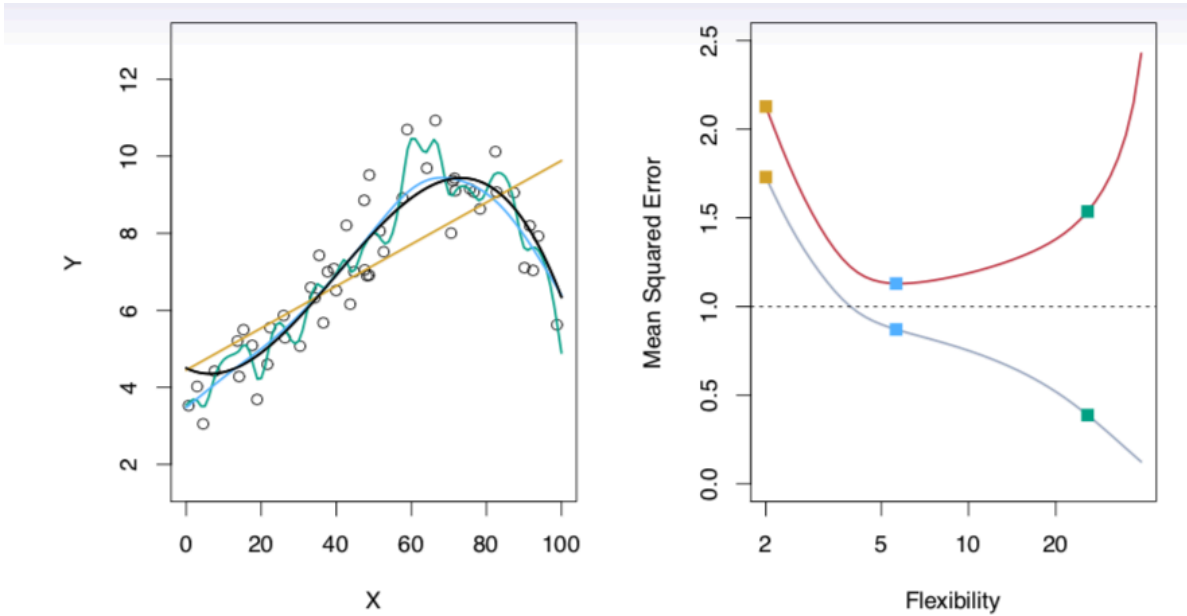


Training set:
0%misclassification

Test set:
8/25=24%
misclassification



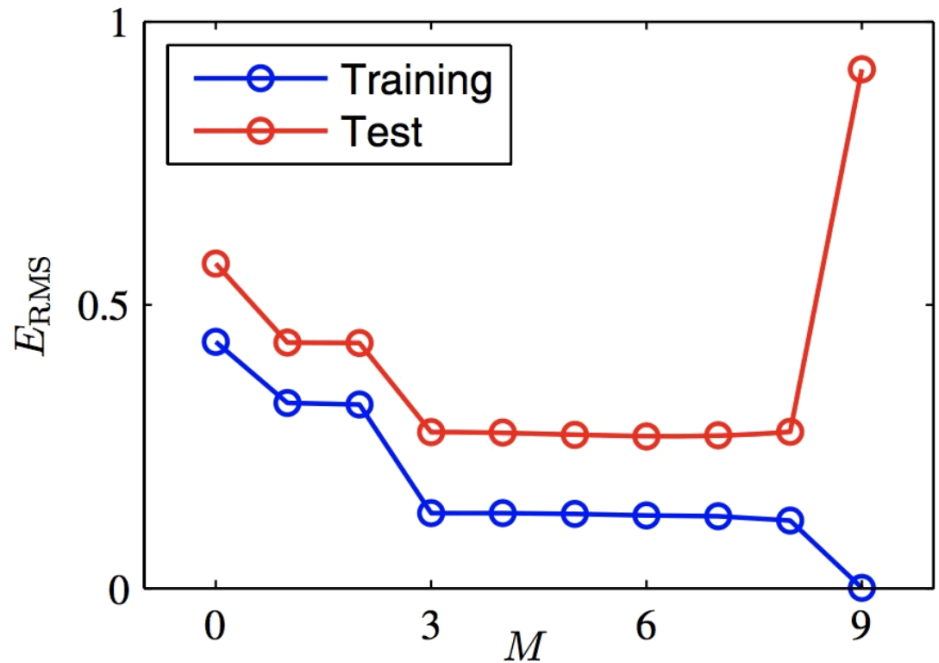
Some examples from regression



Black curve is truth. Red curve on right is MSE_{Te}, grey curve is MSE_{Tr}. Orange, blue and green curves/squares correspond to fits of different flexibility.

Taken from ISLR

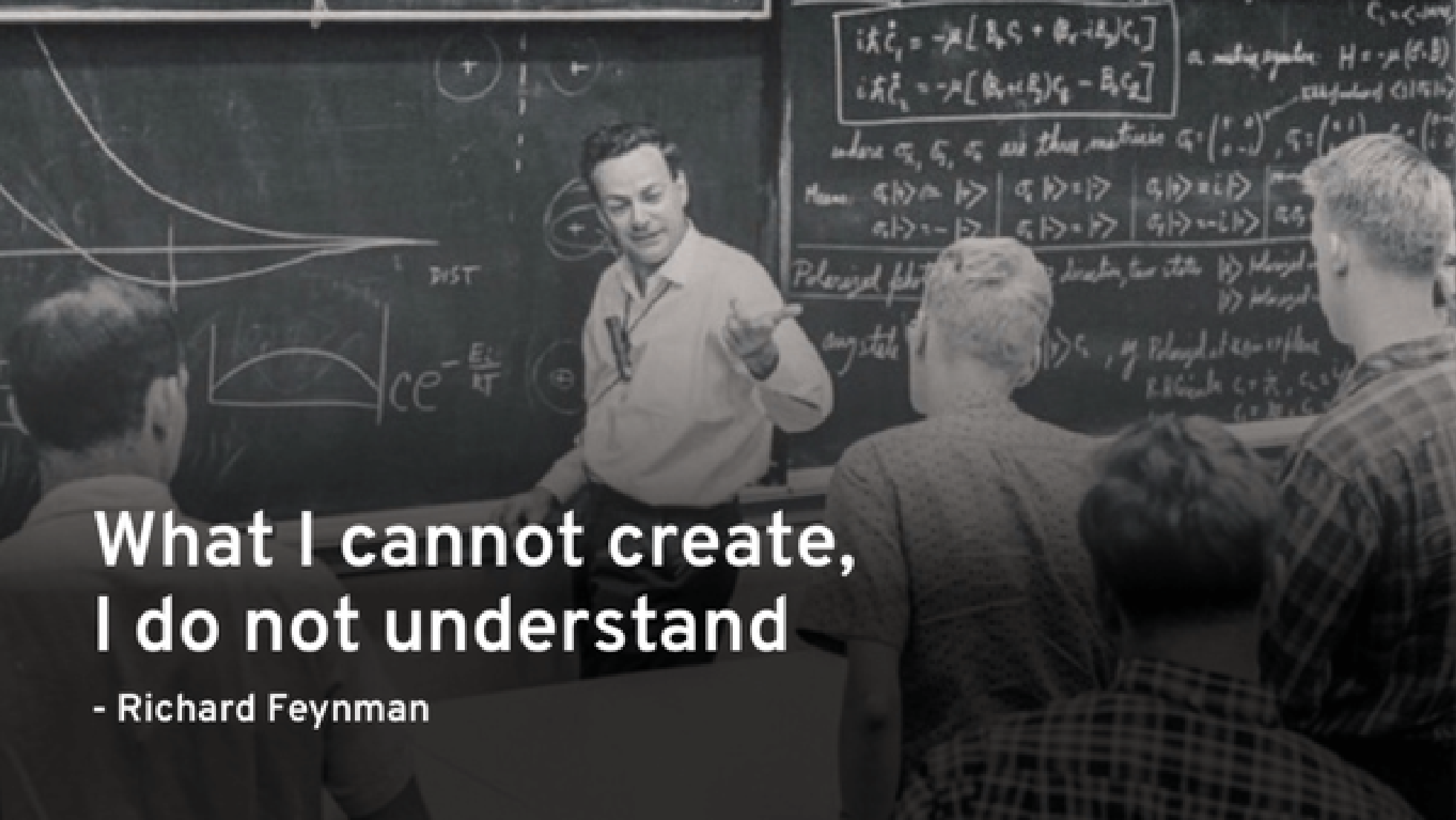
Over- and Underfitting: Intuition



Too simple model
Underfitting

Too complex model
Overfitting

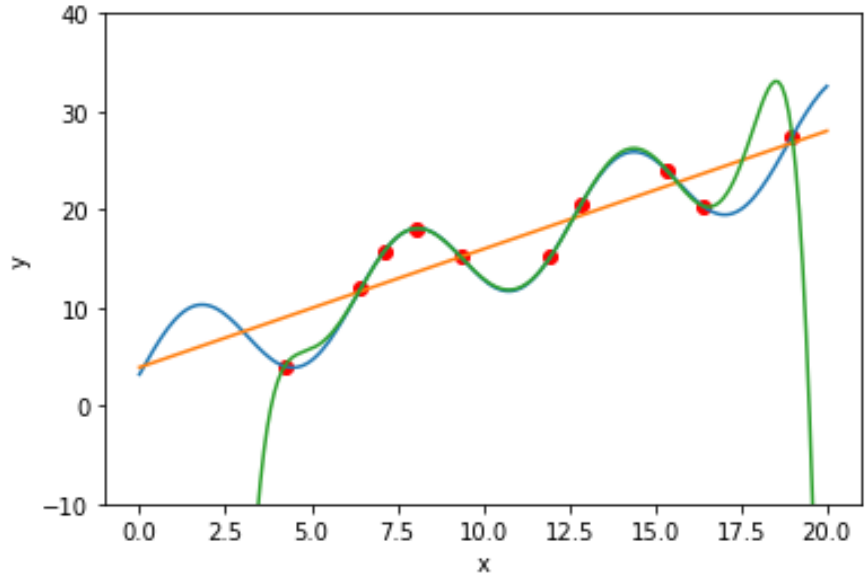
The Bias and Variance in Detail



What I cannot create, I do not understand

- Richard Feynman

Which model will yield the better predictions?



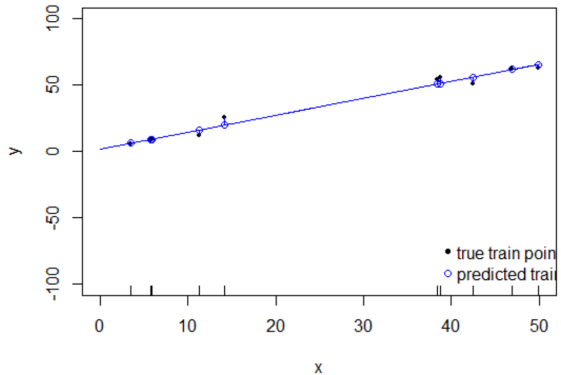
Demo Time

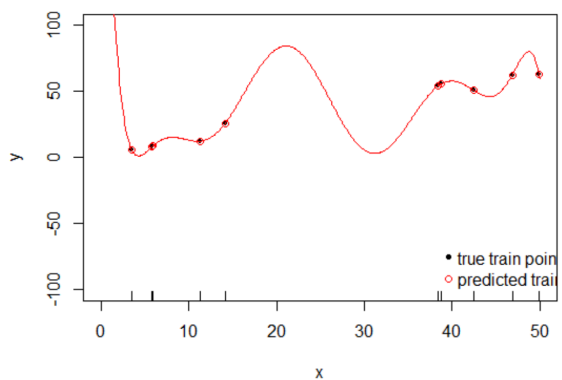
"What I cannot simulate, I do not understand"

Simulation study: https://github.com/ioskn/mldl_htwg/blob/master/bias_variance/Bias_Variance_Sim.ipynb

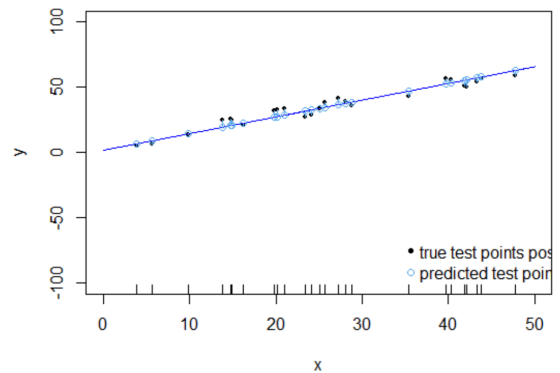
Run 1

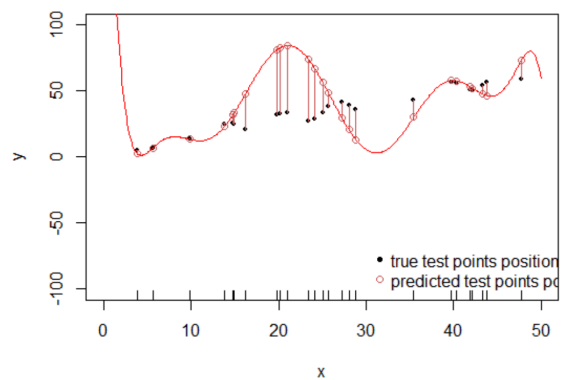
Check performance on **train** data





Check performance on **test** data



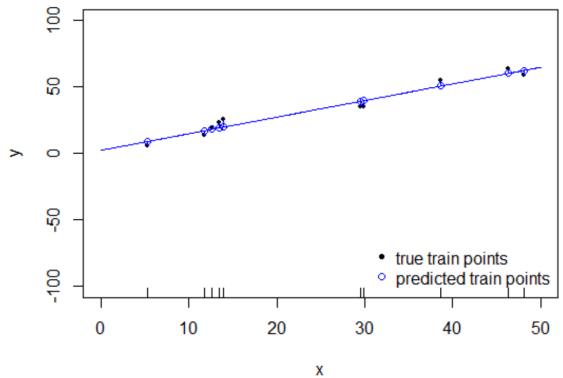


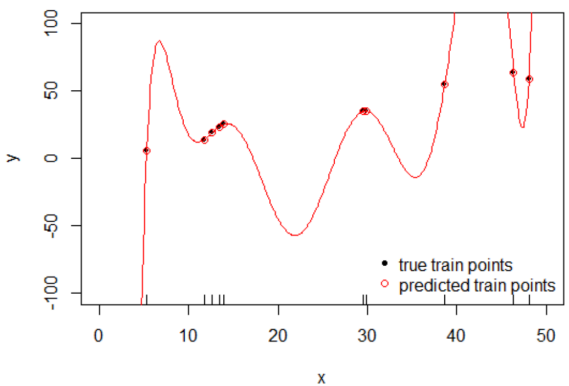
Note: The flexible “overfits” the train data: its performance goes down on test data

The rigid model often “underfits” the data: the true underlying relationship is more complex.

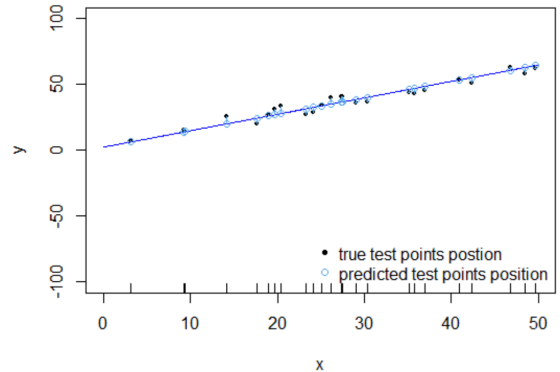
Run 2

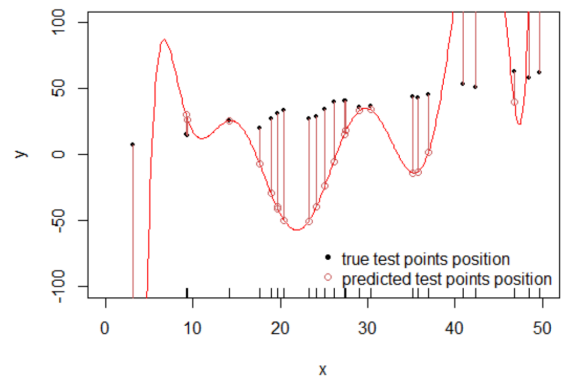
Check performance on **train** data





Check performance on **test** data

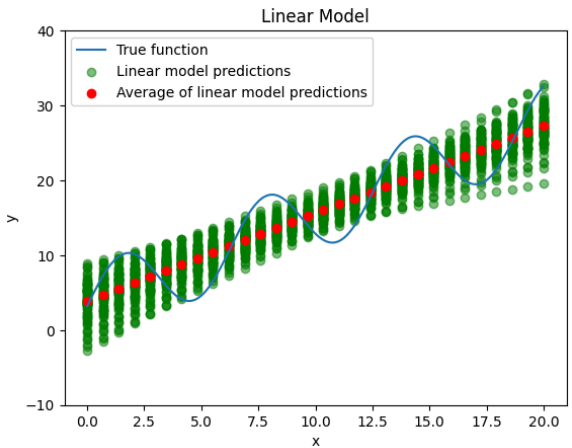




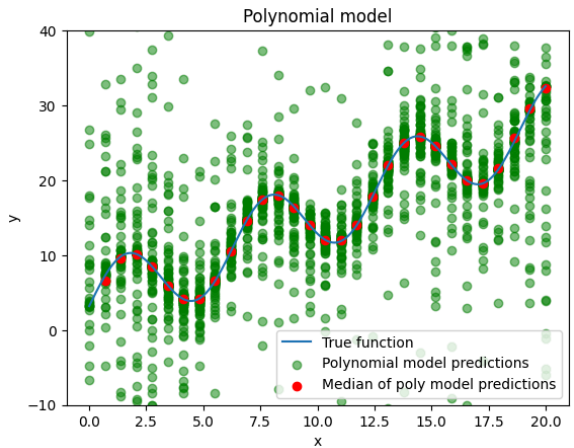
Note: The flexible model is on new train data very different to the run we saw before, the rigid model is very similar

Variance/Bias trade-off of flexible/rigid models

- Simulated data came from an ascending oscillating curve – see grey curve.
- 200 Runs
  - Sample Train set of 10 points and fit both models
  - Evaluate test set at 30 pre-selected x-positions (see tiny points) mean fat points.

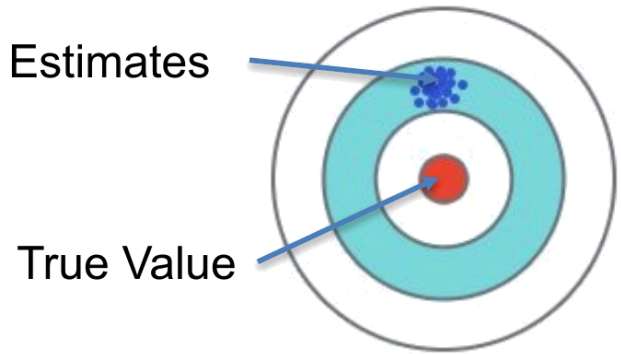


Underfitting
Low Variance
Some bias

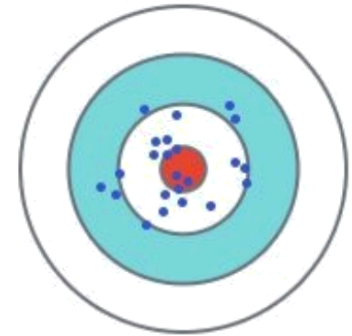


Overfitting
High Variance
Small Bias

Two possible ways to be wrong (bias vs. variance)



Systematically off
(Bias)

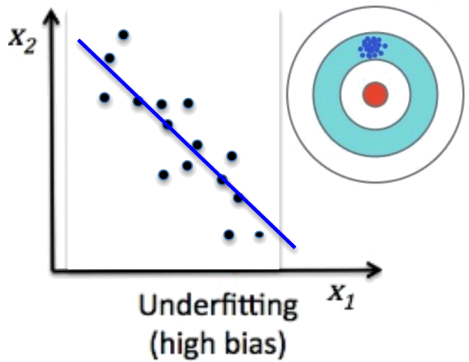


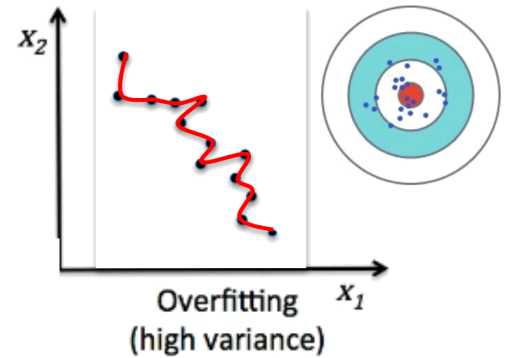
Large fluctuations
(Variance)

Bias and variance of estimates (stuff with hat) for any estimator for example:

- The prediction $\hat{f}(X)$ compared to $f(X)$
- The parameter $\hat{\beta}$ compared to β

The concept of Bias and Variance





A under fitting model

- Is **not flexible enough** for true data structure
- Some systematic error since the model assumes a too simple relationship (high bias)
- Will **not vary a lot if fitted to new train data** (low variance)

A overfitting model

- Is **too flexible for data structure**
- Few errors on train set
- Non-systematic test errors (low bias)
- Will **vary a lot if fitted to new train** data (high variance)

Later Ensemble Methods will use this inside