Module 5: Resampling

TMA4268 Statistical Learning V2023

Stefanie Muff, Department of Mathematical Sciences, NTNU

February 6 and 9, 2023

Acknowledgements

- A lot of this material stems from Mette Langaas and her TAs. I would like to thank Mette for the permission to use her material!
- Some of the figures and slides in this presentation are taken (or are inspired) from James et al. (2013).

Introduction

Learning material for this module

- James et al (2021): An Introduction to Statistical Learning, Chapter 5.
- All the material presented on these module slides.

Additional material for the interested reader: Chapter 7 (in particular 7.10) in Friedman et al (2001): Elements of Statistical learning.

What will you learn?

- What is model assessment and model selection?
- Ideal solution in a data rich situation.
- Cross-validation and what is best:
    - validation set
    - leave-one-out cross-validation (LOOCV)
    - k-fold CV
- Bootstrapping - how and why.

Performance of a learning method

- Our models are “good” when they can generalize.
- We want a learning method to perform well on new data (low test error).
- Inference and understanding of the true pattern (in contrast to overfitting)

This is important both for

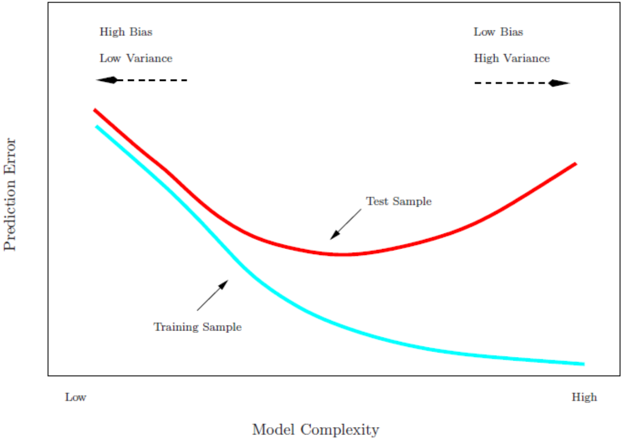
Model selection

Estimate the *performance* of different models to *choose the best model.*

Model assessment

Estimating the performance (prediction error) of the final model, on new data.

Training vs Test Error



Loss functions

In order to define how we measure error, we must first decide for a **loss function**. Here we use:

- *Mean squared error* (quadratic loss) for regression problems (continuous outcomes) $Y_i = f(x_i) + \varepsilon_i$, $i = 1, \dots, n$:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \ .$$

- *Misclassification rate* (0/1 loss) for classification problems where we classify to the class with the highest probability $P(Y = j \mid x_0)$ for $j = 1, \dots, K$:

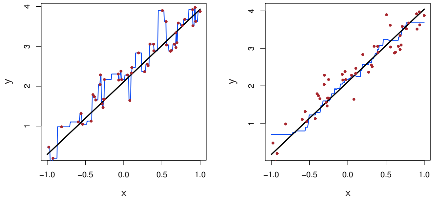
$$\frac{1}{n}\sum_{i=1}^{n}\text{I}(y_i \neq \hat{y}_i) \ .$$

KNN regression (chapter 3.5 in course book)

- The KNN regression method provides a prediction at a value x_0 by finding the closest K points (Euclidean distance) and calculating the average of the observed y values at the points in the respective neighborhood $\mathcal{N}_0$

$$\hat{f}(x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i \ .$$

Illustration: Linear regression with $K = 1$ (left) and $K = 9$ (right).

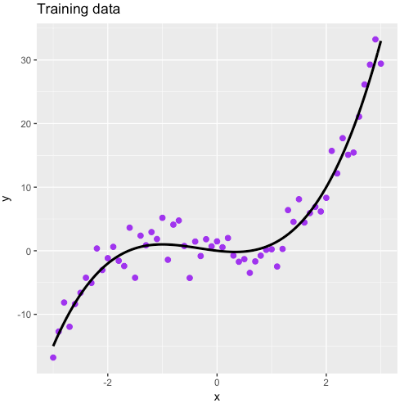


(Figure 3.17 from James et al. (2013)).

What happens for K = number of data points?

Example

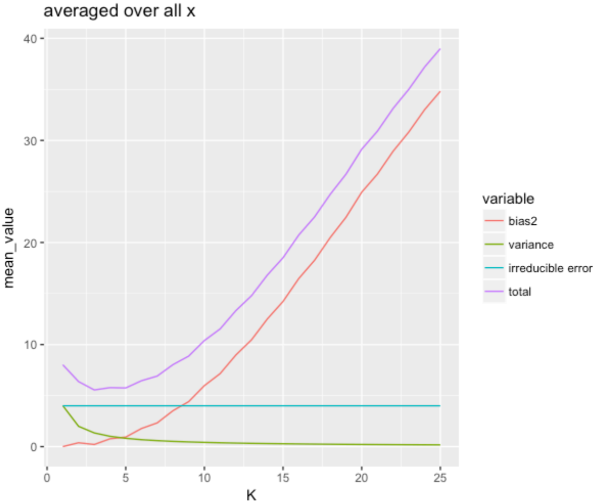
We aim to do *model selection* in KNN-regression, where true curve is $f(x) = -x + x^2 + x^3$ with $x \in [-3, 3]$. $n = 61$ for the training data.



- We have considered $K = 1, \dots, 25$, and repeated the experiment $M = 1000$ times (that is, M versions of training and test set).

Remember: The bias-variance trade-off

For KNN: K small = high complexity; K large = low complexity.



The challenge

- In the above examples we knew the truth, so we could assess training and test error.
- In reality this is of course not the case.
- We need approaches that work with real data!

The data-rich situation (often unrealistic)

If we had a large amount of data we could divide our data into three parts:

- **Training set**: to fit the model
- **Validation set**: to select the best model (*model selection*)
- **Test set**: to assess how well the model fits on new independent data (*model assessment*)

Q: Before we had just training and test. Why do we need the additional validation set?
A: We have not discussed model selection before.

Q: Why can't we just use the training set for training, and then the test set both for model selection and for model evaluation?
A: We will be too optimistic if we report the error on the test set when we have already used the test set to choose the best model.

- If you have a lot of data – great – then you do not need Module 5.
- But, this is very seldom the case – so we will study other solutions based on efficient sample reuse with *resampling* data.
- An alternative strategy for model selection (using methods penalizing model complexity, e.g. AIC or lasso) is covered in Module 6.

We will look at *cross-validation* and the *bootstrap*.

Cross-validation (CV)

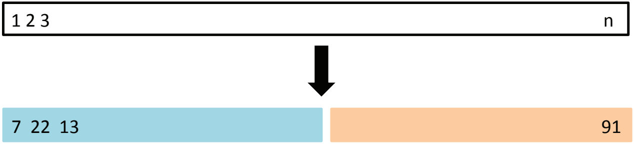
“Model selection” situation: We assume that test data is available (and has been put aside), and we want to use the rest of our data to find the model that performs “best”, that is, *with lowest test error*.

This can be done by:

- the validation set approach (not strictly a *cross*-validation approach).
- leave one out cross-validation (LOOCV).
- k-fold cross-validation (CV), typically $k = 5$ or 10.

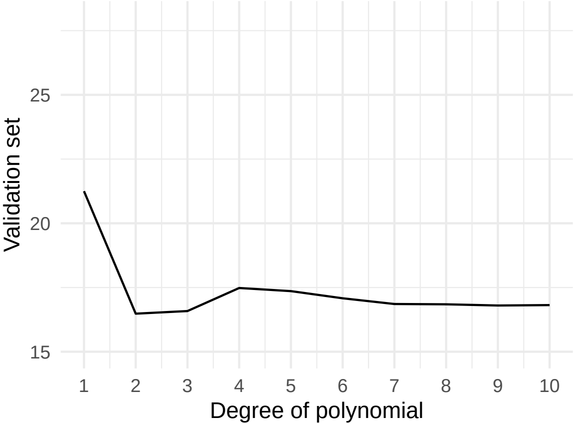
The validation set approach

- Consider the case when you have a data set consisting of n observations.
- To fit a model and to evaluate its predictive performance you randomly divide the data set into two parts ($n/2$ sample size each):
  - a *training set* (to fit the model) and
  - a *validation set* (to make predictions of the response variable for the observations in the validation set)

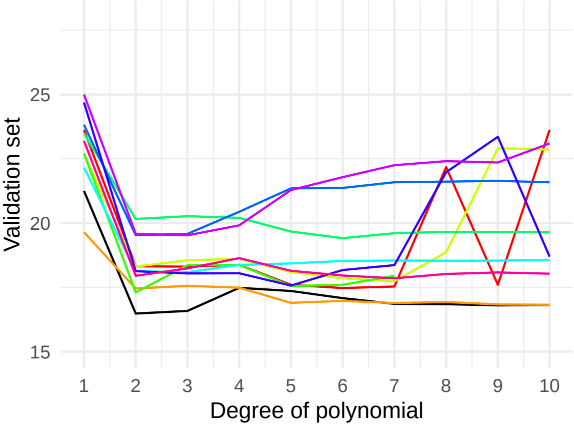


Example of validation set approach

Auto data set (library `ISLR`): predict `mpg` (miles pr gallon) using polynomial function of `horsepower` (of engine), $n = 392$. What do you see?



But what if we select another split into two parts? Many splits:



→ No consensus which model really gives the lowest validation set MSE.

Drawbacks with the validation set approach

- *High variability* of validation set error due to dependency on the set of observation included in the training and validation set.
- *Smaller sample size* for model fit, as only half of the observations are in the training set. Therefore, the validation set error may tend to overestimate the error rate on new observations for a model that is fit on the full data set (the more data, the lower the error).

Better ideas?

Leave-one-out cross-validation (LOOCV)

Leave-one-out cross-validation (LOOCV) addresses the limitations of the validation set approach.

Idea:

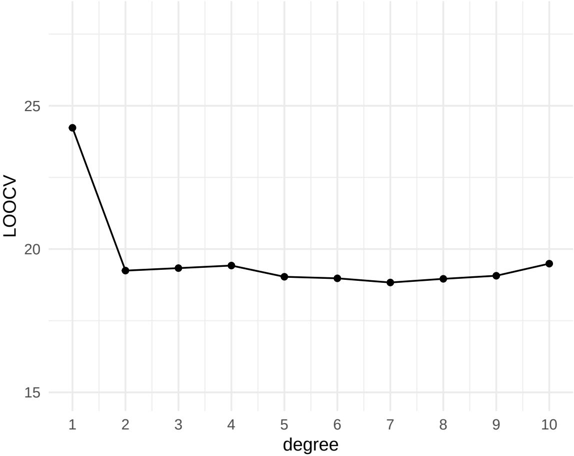
- Only **one observation at a time** is left out (test set size $n = 1$).
- The remaining $n - 1$ observations make up the training set.
- The procedure of model fitting is repeated n times, such that each of the n observations is left out once. In each step, we calculate

$$\mathrm{MSE}_i = (y_i - \hat{y}_i)^2 \ .$$

- The **total prediction error** is the mean across these n models

$$\mathrm{CV}_n = \frac{1}{n} \sum_{i=1}^{n} \mathrm{MSE}_i \ .$$

Regression example: LOOCV



Issues with leave-one-out cross-validation

- Pros:
  - No randomness in training/validation splits!
  - Little bias, since nearly the whole data set used for training (compared to half for validation set approach).
- Cons:
  - Expensive to implement – need to fit n different models.
  - High variance since: two training sets only differ by one observation, thus estimates from each fold highly correlated, which can lead to high variance in their average*.

* Recall that

$$\mathrm{Var}(\sum_{i=1}^{n} a_i X_i) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \mathrm{Cov}(X_i, X_j)$$

$$= \sum_{i=1}^{n} a_i^2 \mathrm{Var}(X_i) + 2\sum_{i=2}^{n}\sum_{j=1}^{i-1} a_i a_j \mathrm{Cov}(X_i, X_j).$$

LOOCV for multiple linear regression

There is a nice shortcut for LOOCV in the case of linear regression:

$$\text{CV}_n = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_{ii}}\right)^2 \ ,$$

where h_i is the ith diagonal element (leverage) of the hat matrix $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, and $\hat{y}_i$ is the ith fitted value from the original least squares fit.

$\rightarrow$ Need to fit the model only once!

See Compulsory exercise 1.

k-fold cross-validation

To address the drawbacks of LOOCV, we can leave out not just one single observation in each iteration, but $1/k$-th of all data.

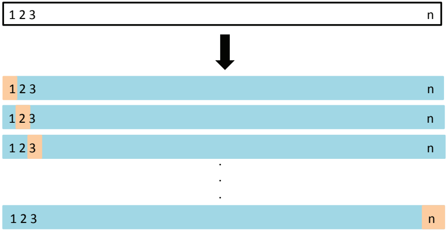
Procedure:

- Split the data into k (more or less) equal parts.
- Use $k-1$ parts to fit and the kth part to validate.
- Do this k times and leave out another part in each round.

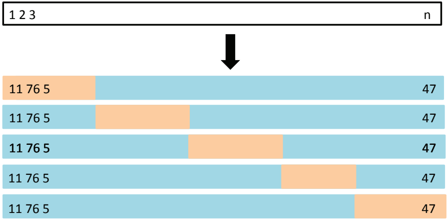
The MSE is then estimated in each of the k iterations $(\text{MSE}_1, \dots, \text{MSE}_k)$. The k-fold CV is then a (weighted) avearge over the k MSEs.

Comparison of LOOCV and *k*-fold CV:

LOOCV:



k-fold:



Formally

- Indices of observations - divided into k folds: $C_1, C_2, \dots, C_k$.
- n_j elements in fold j. If n is a multiple of k then $n_j = n/k$ for all folds.

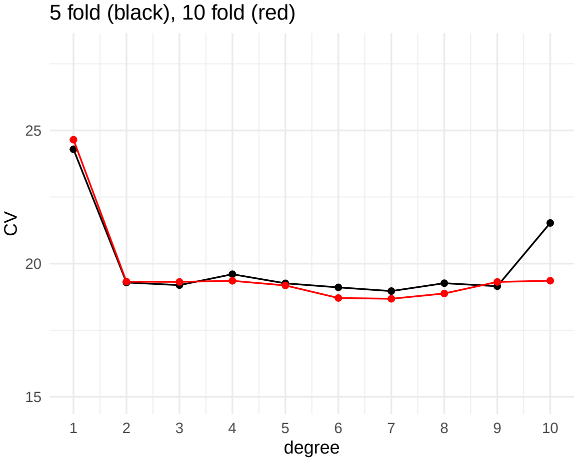
$$\mathrm{MSE}_j = \frac{1}{n_j} \sum_{i \in C_j} (y_i - \hat{y}_i)^2$$

where $\hat{y}_i$ is the fit for observation i obtained from the data with part j removed.

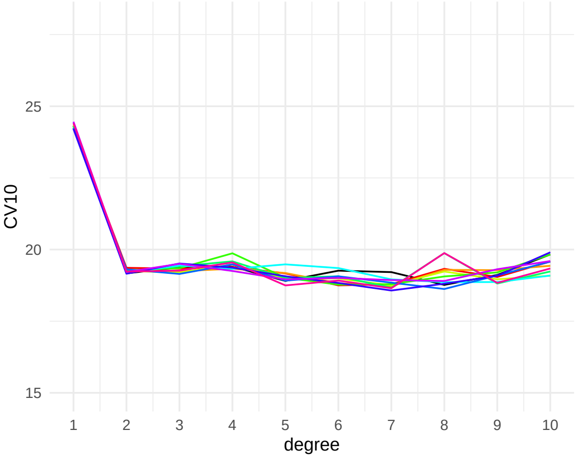
$$\mathrm{CV}_k = \frac{1}{n} \sum_{j=1}^{k} n_j \mathrm{MSE}_j$$

Observe: setting $k = n$ gives LOOCV.

Regression example: 5 and 10-fold cross-validation



10 reruns (different splits) of the 10-CV method - to see variability:



There still *is* variability, but *much less* than for validation set approach.

Issues with k-fold cross-validation

1. The *result may vary* according to how the folds are made, but the variation is in general lower than for the validation set approach.
2. Computational load lower with $k = 5$ or 10 than LOOCV.
3. The training set is $(k-1)/k$ times the size of the original data set - the estimate of the prediction error is biased upwards.
4. This bias is the smallest when $k = n$ (LOOCV), but we know that LOOCV has high variance.
5. Due to the *bias-variance-trade-off*, k-fold CV often gives more accurate estimates of the test error rate than does LOOCV.
   $\rightarrow k = 5$ or $k = 10$ is used as a compromise.

Choosing the best model

- There is a model parameter (maybe K in KNN or the degree of the polynomial), say θ, involved to calculate CV_j, $j = 1, \dots, k$
- Based on the CV vs θ-plot we can choose the model with *the smallest* CV_k as our best model.
- We then fit this model using the whole data set (not the test part, that is still kept away), and evaluate the performance on the test set.

One standard error rule:

Denote by $\mathrm{MSE}_j(\theta)$, $j = 1, \dots, k$ the k parts of the MSE that together give the CV_k.

We can compute the sample standard deviation (standard error) of all $\mathrm{MSE}_j(\theta)$, $j = 1, \dots, k$

$$\widehat{\mathrm{SE}}(\mathrm{CV}_k(\theta)) = \sqrt{\sum_{j=1}^{k} (\mathrm{MSE}_j(\theta) - \overline{\mathrm{MSE}}(\theta))/(k-1)}$$

for each value of the complexity parameter θ.[1]

The *one standard error rule* is to choose the simplest model (*e.g.*, with lowest polynomial degree) within one standard error of the minimal error.

[1] Strictly speaking, this estimate is not quite valid. Why?

k-fold cross-validation in classification

What do we need to change from our regression set-up?

- For LOOCV $\hat{y}_i$ is the fit for observation i obtained from the data with observations i removed, and $\text{Err}_i = I(y_i \neq \hat{y}_i)$. LOOCV is then

$$\text{CV}_n = \frac{1}{n} \sum_{i=1}^{n} \text{Err}_i$$

- The k-fold CV is defined analogously.
- Chapter 5.1.5 in the course book.