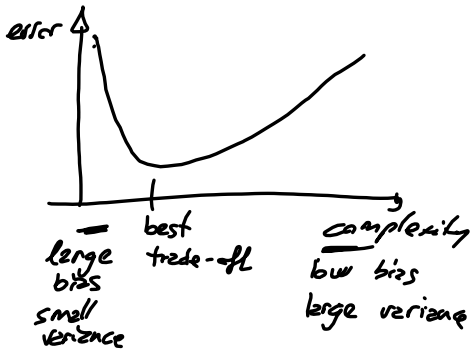
Evaluate the model / find the best complexity

$$E[(f(x) - \hat{f}(x))^2]$$

$\text{bias}^2 + \text{variance}$

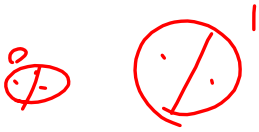


→ we do not know $f(x)$

→ find a way to estimate the expected prediction error with the data we have

1) random split in training and test set

training	test
50%	50%
66.6%	$33.\overline{3}$ %
75%	25%

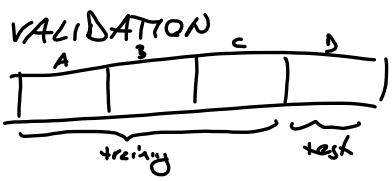


→ when we have 0-1 response, what happens if the response is very unbalanced → stratified ~~re~~sampling
→ perform the ~~random~~ split separately for case (1) and controls (0)

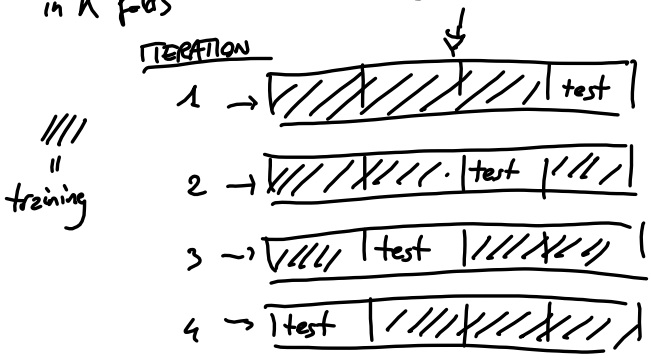
- possible issues:
  → it can perform badly if we do not have enough data (not really common in data science, where usually the problem is that the datasets are too big);
  → if we only perform a split, we are not fully using all the information in the data (we do not use the test set to any training part) (we only evaluate on a specific subset (test set) of the data)

2) → CROSS-VALIDATION

data split in K folds →



$K=4$ $\quad err = \frac{1}{\#\text{test}} \sum_{i \in \text{test}} (y_i - \hat{f}^{\text{train}}_{(x_i)})^2$



$$err_1 = \frac{1}{|D|} \sum_{i \in D} (y_i - \hat{f}^{A,B,C}_{(x_i)})^2$$

$$err_2 = \frac{1}{|C|} \sum_{i \in C} (y_i - \hat{f}^{A,B,D}_{(x_i)})^2$$

$$err_3 = \frac{1}{|B|} \sum_{i \in B} (y_i - \hat{f}^{A,C,D}_{(x_i)})^2$$

$$err_4 = \frac{1}{|A|} \sum_{i \in A} (y_i - \hat{f}^{B,C,D}_{(x_i)})^2$$

$$err_{cv} = \frac{1}{4} \sum_{i=1}^{4} err_i$$

y_D	$\hat{y} \leftarrow \hat{f}^{A,B,C}_{(x_i)}$	
1	0.4 → 0	1
0	0.7 → 1	1
1	0.5 → 1	0

$\frac{1}{3}(1+1+0) = 0.\overline{66}$

$x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$

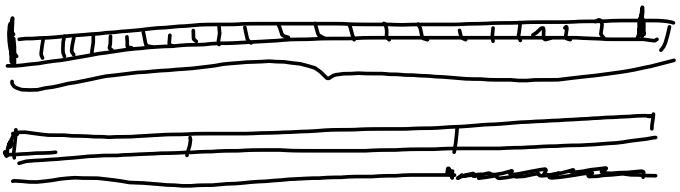
→ A: x_6, x_2 | B: x_1, x_8 | C: x_4, x_3 | D: x_5, x_7

→ A: x_5, x_1 | B: x_6, x_7 | C: x_2, x_5 | D: x_8, x_3

The number of folds is arbitrary (K can be every number $2 \leq K \in n$), and therefore the procedure is called K-fold cross-validation
→ use in turn 1 of the K folds as a test set, in which we evaluate the models trained on the other K-1 folds

Usually we use $K=5$, $K=10$, $K=n$
↳ leave-one-out cross-validation

The choice of K also follow a bias-variance trade-off idea
→ larger K ⇒ less bias, more variance
→ smaller K ⇒ more bias, less variance
→ master course STK-IN4300 we will investigate this in details



→ larger K makes the required computational time larger (due to increased number of iterations)
↳ exception: in the case of linear predictor, there is a closed form for the leave-one-out cross-validation

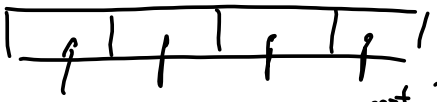
$$y_i - \hat{y}_{-i} = \frac{(y_i - \hat{y}_i)}{1 - p_{ii}} \quad \text{(computed } \hat{f}_{(x_i)} \text{ all points, including } i\text{)} \Rightarrow \text{err}_{\text{loocv}} = \frac{1}{N} \sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{(1 - p_{ii})^2}$$

($\hat{f}_{(x_i)}$: fitting set i)

Note:
- loocv is a deterministic procedure
- the other K-fold procedure are random (because they are based on a random split)
  → for a practical/computational point of view, set the seed (in R, set.seed(#)) in order to obtain reproducible results.

$x_1 - x_8$

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8



→ repeated cross-validation

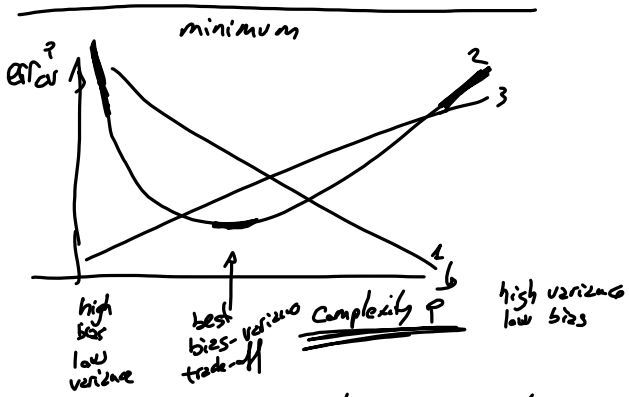
repeat B times
1) split observations in K folds
2) perform K-fold cross-validation $\text{err}_{cv}^{[b]} = \frac{1}{K} \sum_{i=1}^{K} \text{err}_i^{[b]}$

average the results $\text{err}_{\text{repCV}} = \frac{1}{B} \sum_{b=1}^{B} \text{err}_{cv}^{[b]}$

When we want to dermine the "best" complexity for a model
→ split data in k-fold
→ train all models (poly(1), poly(2), ...) on k-1 folds
→ test all of them in the remaining fold

test set	p=1	p=2	p=3	...	p=20
fold 1	$err_1^{p=1}$	$err_1^{p=2}$			
fold 2	$err_2^{p=1}$				
⋮	⋮				
fold k	$err_k^{p=1}$				
average	$err_{cv}^{p=1}$	$err_{cv}^{p=2}$			$err_{cv}^{p=20}$

← $p = \underset{p}{\text{argmin}}\ err_{cv}^{p}$



Alternative to cross-validation: bootstrapping

based on non-parametric bootstrap:

1) generate a bootstrap sample, by sampling with replacement n observations from the original sample
e.g. $X = \{x_1, x_2, x_3, x_4, x_5\} \longrightarrow X^* = \{x_2, x_3, x_3, x_4, x_2\}$
2) fit the model on the bootstrap sample → $\hat{f}^*(x)$
3) use the observation not included on the bootstrap sample as a test set
e.g. $X_{test} : \{x_1, x_5\}$
e.g. $err_b = \frac{1}{2}\left[(y_1 - \hat{f}^*(x_1))^2 + (y_5 - \hat{f}^*(x_5))^2\right]$
4) repeat 1 to 3 B times and average the results

$$err_{boot} = \frac{1}{B}\sum_{b=1}^{B} err_b$$

advantage: our training set has the same sample size of the dataset
→ theoretical properties that we will see in STK-IN 4300

note: bootstrapping is overestimating the error
⇒ more complex procedures (variants) that take into account this
- 0.632 bootstrapping
- 0.632+ "

It is important to use as a test set only observations that do not belong to the bootstrap sample

1) generate bootstrap sample x^*
2) fit the model $\hat{f}^*(x)$
3) compute the error on the dataset $err_b = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}^*(x_i))^2$
4) repeat B times and average errors

1) split in training and test
2) cross-validation / bootstrapping
3) criteria based on information

We saw:
- the typical statistical approach for parameter estimation is based on the maximization of the log-likelihood
- we cannot directly rely on log-likelihood maximization to select the right model → more complex model, higher log-likelihood → overfitting

Idea:
- add a penalty term to penalize for too complex model
  → not maximize a log-likelihood, but a penalized log-likelihood
  (minimize) (deviance)
  → $IC = -2\,\text{log-likelihood} + \text{penalty}(p)$

penalty(p) penalizes for too complex models, and depends obviously on the complexity

We want to minimize IC.

How to specify penalty(p)
→ must be positive and increase with p;
→ the penalty for p parameters must be larger than p

penalty $= \frac{1}{2}p$

$$2\left(\ell(\vartheta_{p+1}) - \ell(\vartheta_p)\right) \overset{H_0}{\sim} \chi^2_1 \qquad H_0: \vartheta_{(p+1)} = 0$$

ϑ_{p+1} is a parameter of dimension $p+1$
ϑ_p " " " " p

$\mathbb{E}[\chi^2_1] = 1$

By adding a term without any significance to the model with ϑ_p, we expect a decrease in terms of $-2\ell(\vartheta)$ of 1

$$IC = -2\,\ell(\hat{\vartheta}) + \text{penalty}(p)$$

Typical choices for penalty(p) are

NAME	penalty(p)
Akaike Information Criterion (AIC)	$2p$
corrected AIC (AIC_c)	$2p + \frac{2p(p+1)}{n-(p+1)}$
Bayesian Information Criterion (BIC)	$p\log(n)$

AIC

Idea: we want to minimize the Kullback-Leiber divergence between the true model and the fitted model

$$KL\left(P_0(y), P(y,\vartheta)\right) = E_{P_0}\left[\log\frac{P_0(y)}{P(y,\vartheta)}\right] = \underbrace{E_{P_0}\left[\log P_0(y)\right]}_{\text{true model}} - \underbrace{E_{P_0}\left[\log P(y,\vartheta)\right]}_{\text{fitted model}}$$

independent of ϑ

we can only act on this part
⇒ the largest, the better
⇓ maximum likelihood

The key point is that the estimation of $E_{P_0}[\log P(y,\vartheta)]$ is

$$\log P(y,\vartheta) - p$$

multiplied by the usual factor -2 → $AIC = -2\log(P(y;\vartheta)) + 2p$
to align that to the standard likehood quantities

Alternative interpretation (Hastie, Tibshirani & Friedman, 2009):

$-2 \log p(y; \hat{\theta}) + 2p$ → correction for (over) optimism

estimation of the training error

STK-IN4300 why

AIC = training error + $\hat{\omega}$

AIC_c is a correction for small sample size

$$2p + \frac{2p(p+1)}{n-(p+1)} \xrightarrow{n \to \infty} 2p$$

When $n \to \infty$, $AIC = AIC_c$

BIC is motivated differently (STK-IN4300)

Consider the two penalties

AIC: $2p$

BIC: $p \log(n)$

$\log n > 2 \quad n > e^2 \approx 7.4$

⇒ less complex model for $n \geq 8$

AIC: better for prediction (when including a non-relevant variable is not too bad as long as the variance is not increased too much)

BIC: support less complex (sparser) models.

In theory, BIC works the best (is consistent)

In practice, AIC works better (BIC tends to select too sparse models)