Model selection

Lecture 2a

Overview

- Model selection
- Model evaluation

An estimator

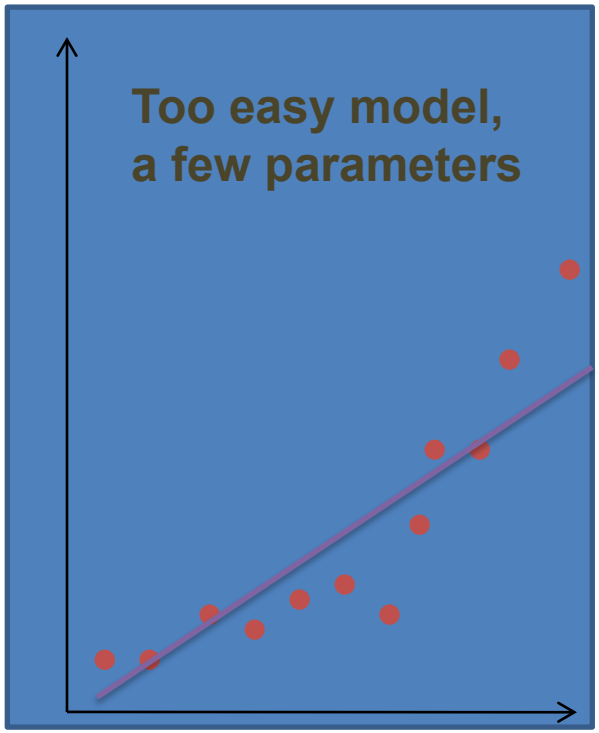
- $\widehat{\boldsymbol{\theta}} = \delta(T)$ (some function of your training data) – an **estimator**
- Optimal parameter values?→there can be many ways to compute them (MLE, shrinkage…)
  - There is no easy way to compare estimators in frequentist tradition

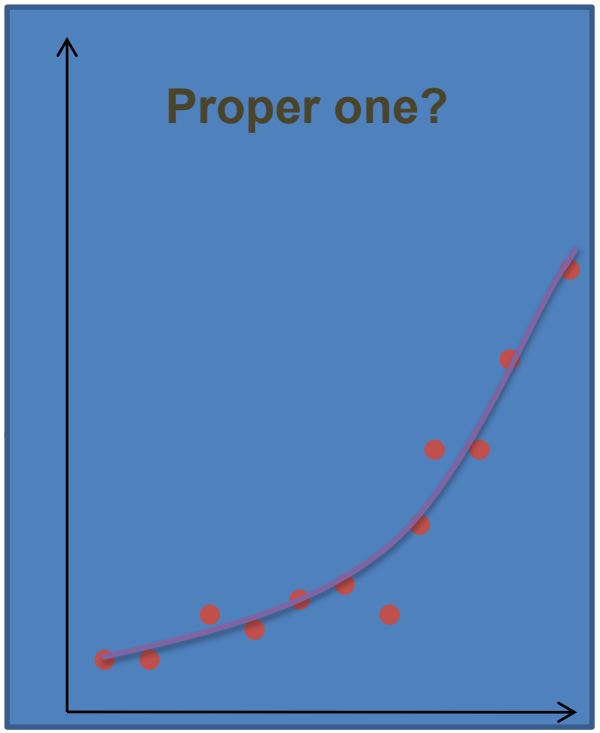
Example: Linear regression

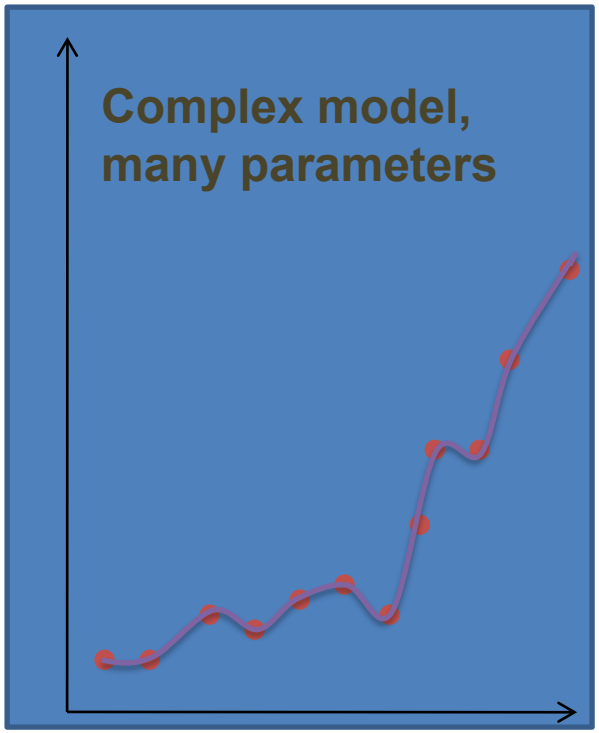
- Estimator 1: $\boldsymbol{\theta} = (X^T X)^{-1} X^T Y$ (maximum likelihood)
- Estimator 2: $\boldsymbol{\theta} = (0, \dots, 0, 1)$
- Which one is better?
  - A comparison strategy is needed!

Overfitting

- Complex model can overfit your data







Model selection

- ML can lead to overfitting
- How can we find appropriate parameter values?
- How can we compare between different models?

Error functions

- Loss functions $L(y, \hat{y})$ used to evaluate the quality of training

- **Error functions** $E(y, \hat{y})$ used to measure the quality of prediction

  - Misclassification error $E(\mathrm{y}, \hat{y}) = \begin{cases} 0, y = \hat{y} \\ 1, y \neq \hat{y} \end{cases}$
  - Squared error $E(y, \hat{y}) = (y - \hat{y})^2$

- Formulas can be same, different purpose

Error vs loss function

- Should error function be same as loss function?
- **Normal practice: Choose the loss related to minus loglikelihood**

Example: Predicting the risk of cancer:

$$E(y = H, \hat{y} = C) = 1$$
$$E(y = C, \hat{y} = H) = 1000$$
$$Loss\ matrix = \begin{pmatrix} 0 & 1000 \\ 1 & 0 \end{pmatrix}$$

- One can show: $\frac{p(y = H|x)}{p(y=C|x)} > 1000 \rightarrow classify\ as\ H$

Data generating process

Data generating process

$p(\boldsymbol{x}, y)$

https://upload.wikimedia.org/wikipedia/commons/thumb/4/4d/Cat_November_2010-1a.jpg/1200px-Cat_November_2010-1a.jpg

$y = "Cat"$

https://www.collinsdictionary.com/images/full/dog_230497594.jpg

$y = "Dog"$

Model selection

- Given a model, choose the optimal parameter values
  - Decision theory

- If we know the true distribution $p(y, \boldsymbol{x})$ then we choose the optimal model by minimizing the **expected risk**:

$$\min_{\hat{y}} \bar{E}_{new} = \min_{\theta} \int_{(x_*, y_*)} \int_{T} E\big(y_*, \hat{y}(\boldsymbol{x}_*, \mathrm{T}, \boldsymbol{\theta})\big) p(\boldsymbol{x}_*, y_*) d\boldsymbol{x}_* dy_*\, dT$$

Model selection

- **Problem**: data generating process is unknown →can not compute expected risk!

- Approximation 1: Instead of considering all possible T, take only **one T→expected new data error**

$$E_{new} = \int_{(x_*, y_*)} E\big(y_*, \hat{y}(\boldsymbol{x}_*, \mathrm{T}, \boldsymbol{\theta})\big) p(\boldsymbol{x}_*, y_*) d\boldsymbol{x}_* dy_*$$

Hold-out method

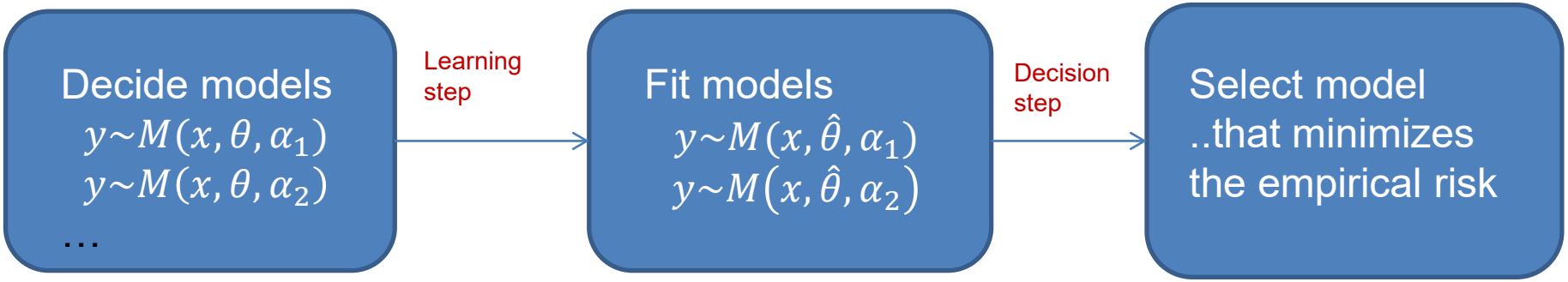
- Fix T as a particular training set
- Approximation 2:
  - $\int_{(x_*,y_*)} E\big(y,\hat{y}(x_*,\mathrm{T},\boldsymbol{\theta})\big)p(\boldsymbol{x}_*,y_*)d\boldsymbol{x}_*dy_* \approx \frac{1}{|V|}\sum_{(\boldsymbol{x}_*,y_*)\in V} E\big(y_*,\hat{y}(\boldsymbol{x}_*,\mathrm{T},\boldsymbol{\theta})\big)$
- **Validation error** (**empirical risk**)

$$E_{hold-out} = \frac{1}{|V|}\sum_{(\boldsymbol{x}_*,y_*)\in V} E(y_*,\hat{y}(\boldsymbol{x}_*,T,\boldsymbol{\theta}))$$

  - Model is learned by Maximum Likelihood using training set
  - Validation error estimated by using validation set
  - Model with minimum validation error is selected
- Note: **training error** can be estimated by replacing V by T

General model selection strategy

- Given data $D = \{\boldsymbol{x}_i, y_i, i = 1 \ldots n\}$

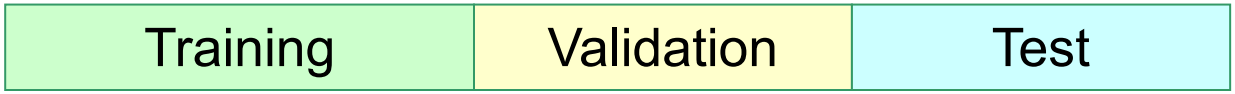


- At learning step, Maximum Likelihood is usually used

- α_i can be different things:
  - Type of distribution
  - Number of variables in the model
  - Regularization parameter value
  - …

Hold-out method

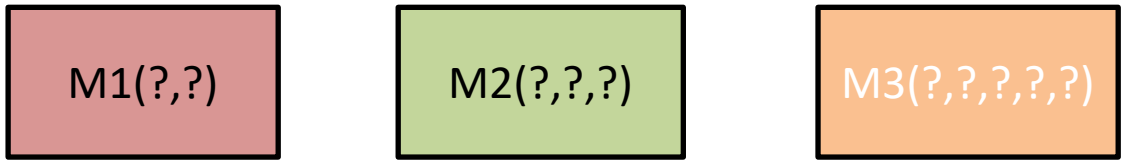
Divide into training, validation and test sets

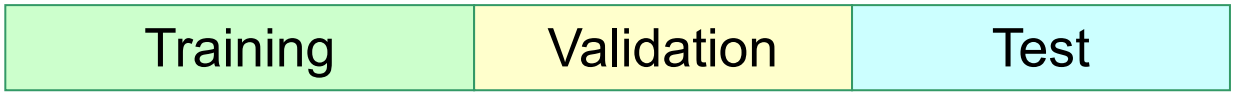


- Choose proportions in some way

Hold-out method

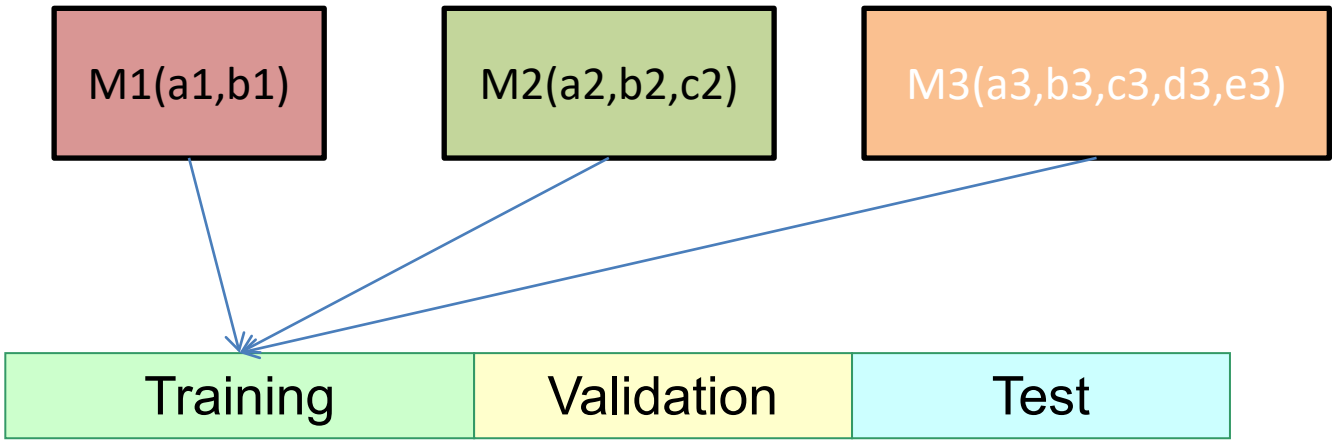
- Given: training, validation, test sets and models to select between





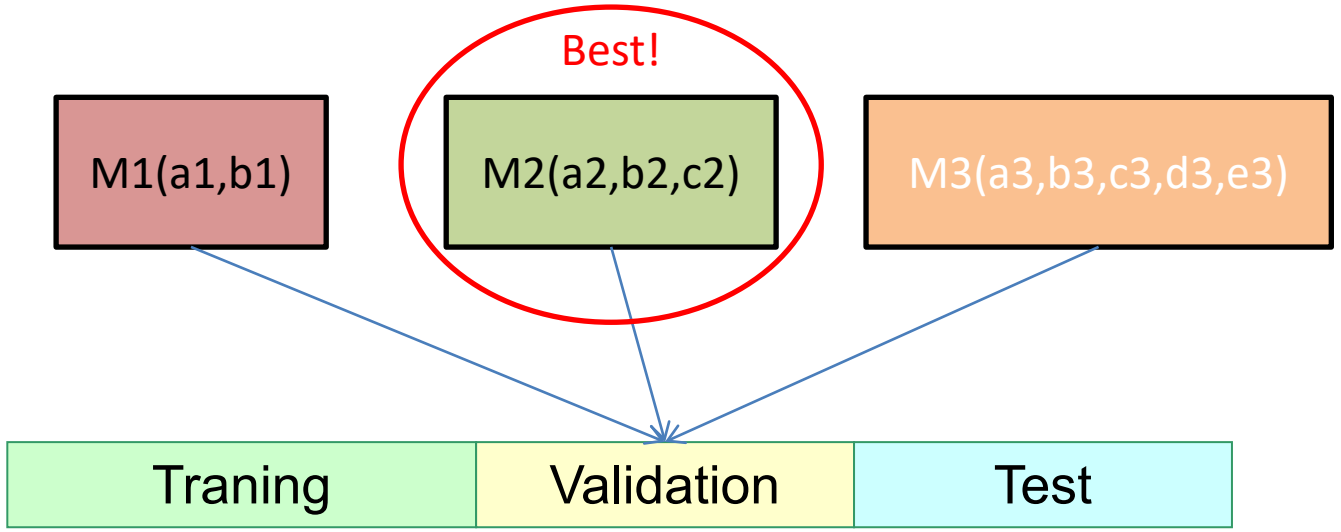
Hold-out method

- Training set is to used for fitting models to the dataset by using given loss function



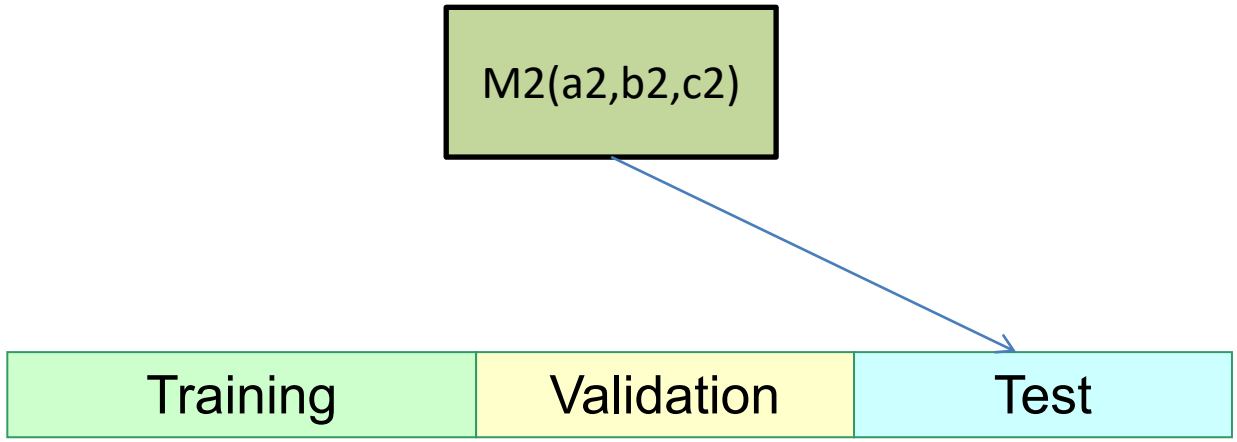
Hold-out method

- Validation set is used to choose the best model (lowest risk)

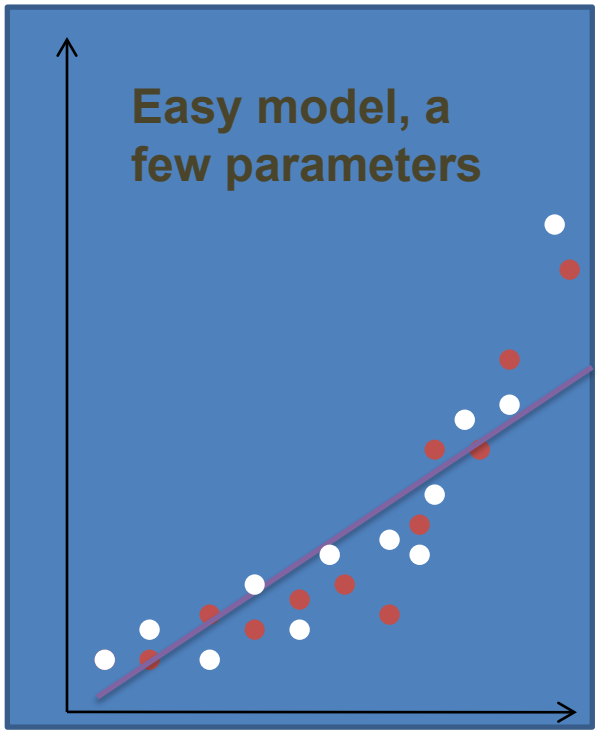


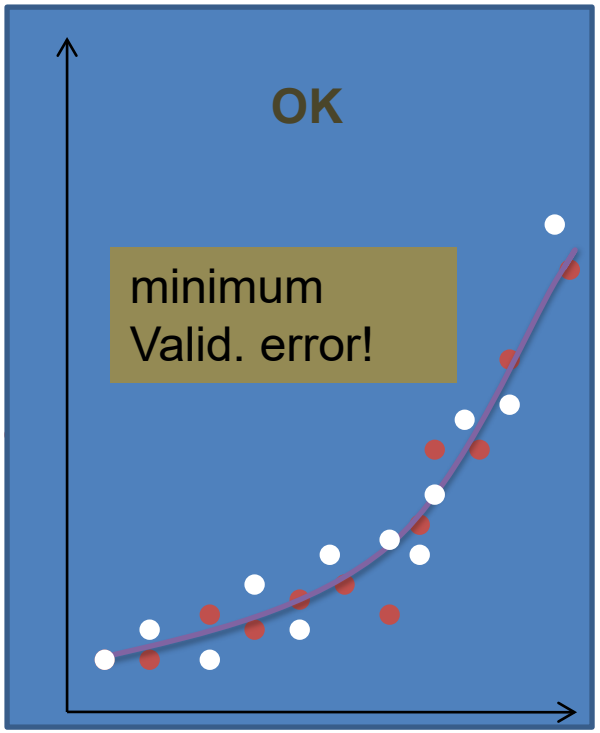
Hold-out method

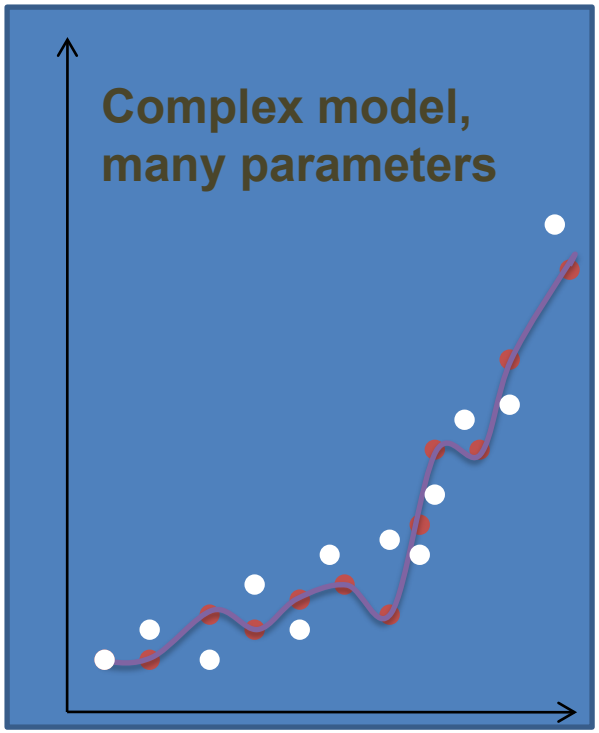
- Test set is used to test a performance on a new data



Hold-out method







Hold-out method: remarks

- Data needs to be shuffled before split
- Method is suitable for large data otherwise training affected
- Proportions: increasing % of training data generally leads to better performance but the quality of ”Approximation 2” decreases

Hold-out in R

- How to partition into train/test?
  - Use set.seed(12345) in the labs to get identical results

```
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.7))
train=data[id,]
test=data[-id,]
```

- How to partition into train/valid/test?

```
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=data[id,]

id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))
valid=data[id2,]

id3=setdiff(id1,id2)
test=data[id3,]
```

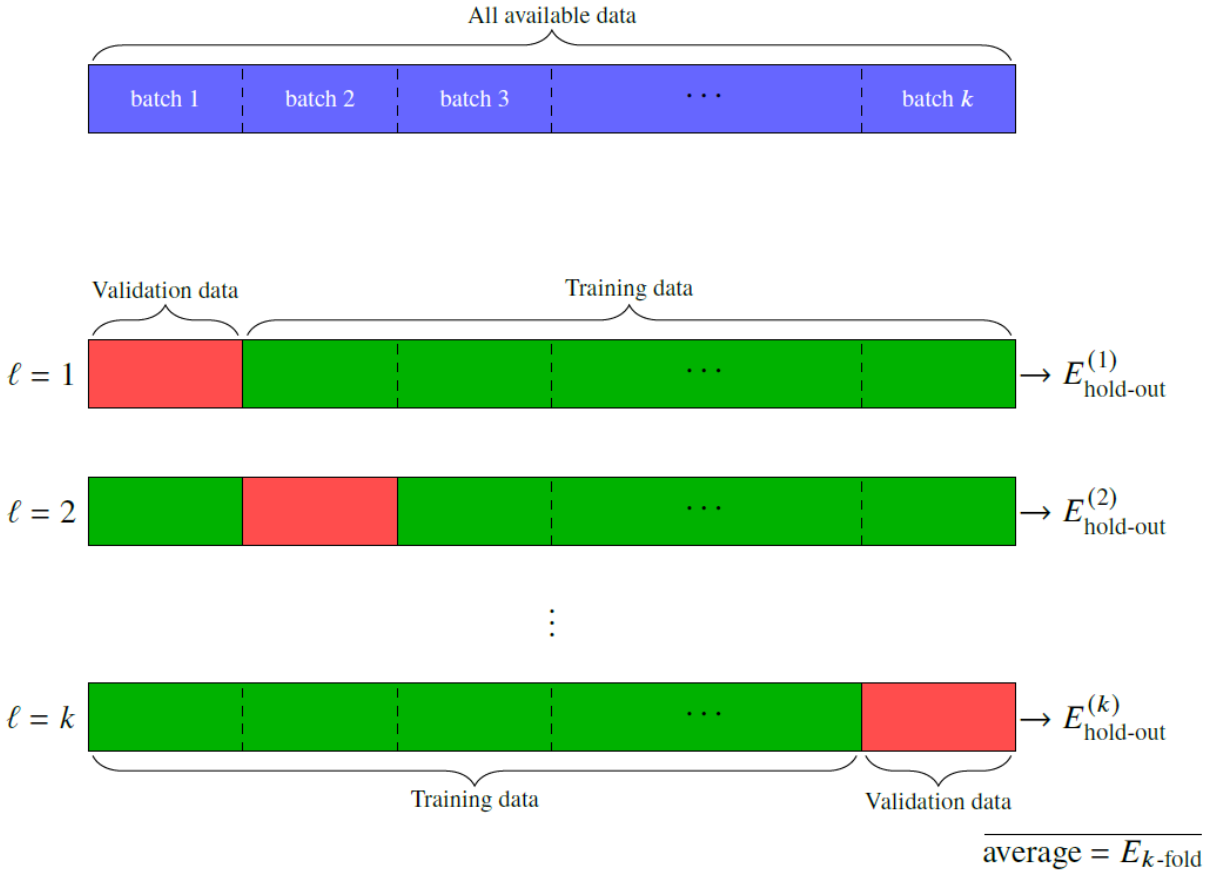
Cross-validation

- Compared to holdout method:
  - Why do we use only some portion of data for training- can we use more?
  - Approximation 1: choose T as **different** subsets of data
  - Approximation 2: choose $(\boldsymbol{x}_*, y_*)$ from the remaining data

Cross-validation K-fold cross-validation (rough scheme, show picture):

1. Permute the observations randomly
2. Divide data-set in K roughly equally-sized subsets
3. Remove subset #i and fit the model using remaining data.
4. Predict the function values for subset #i using the fitted model.
5. Repeat steps 3-4 for different i
6. E_{k-fold}= mean hold out loss

Cross-validation



Cross-validation

Cross-validation

- E_{k-fold} is used as approximation of E_{new}

Note: if $k=n$ then method is *leave-one-out* cross-validation.

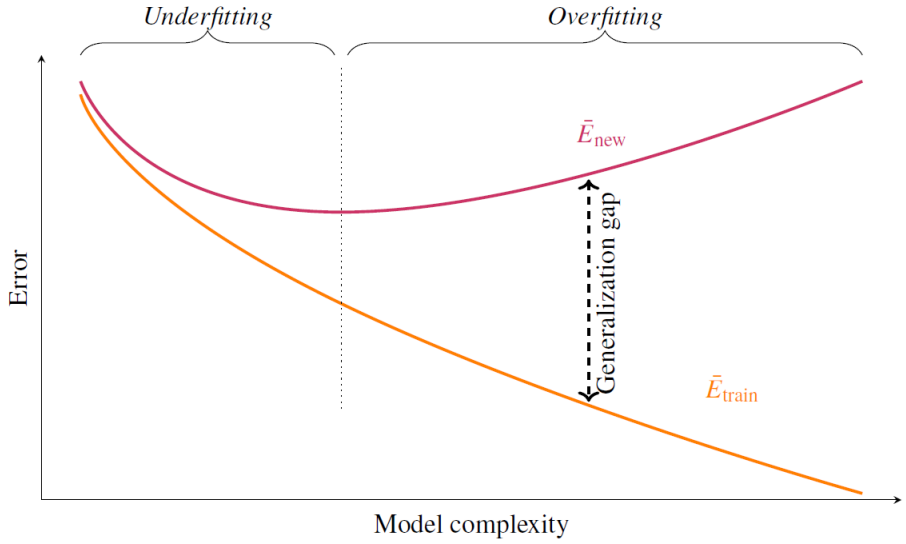
What to do if n is not
a multiple of k?

Cross-validation vs Holdout

- Holdout is easy to do (one model fit)
- Cross validation is computationally demanding (many model fits)
- Holdout is applicable for large data
  - Otherwise, model selection performs poorly
- Cross validation is more suitable for smaller data
- In both cases, **test** set gives unbiased prediction error

Model selection

- Dependence on model complexity
  - $\bar{E}_{new} = \int_{(\boldsymbol{x}_*, y_*)} \int_T E\big(y_*, \hat{y}(\boldsymbol{x}_*, \mathrm{T}, \theta)\big) p(\boldsymbol{x}_*, y_*) d\boldsymbol{x}_* dy_*\, dT$
  - $\bar{E}_{train} = \int_T \int_{(\boldsymbol{x}_*, y_*) \in T} E\big(y_*, \hat{y}(\boldsymbol{x}_*, \mathrm{T}, \theta)\big) d\boldsymbol{x}_* dy_*\, dT$



Model selection

- **Generalization gap**

$$generalization\ gap = \bar{E}_{new} - \bar{E}_{train}$$

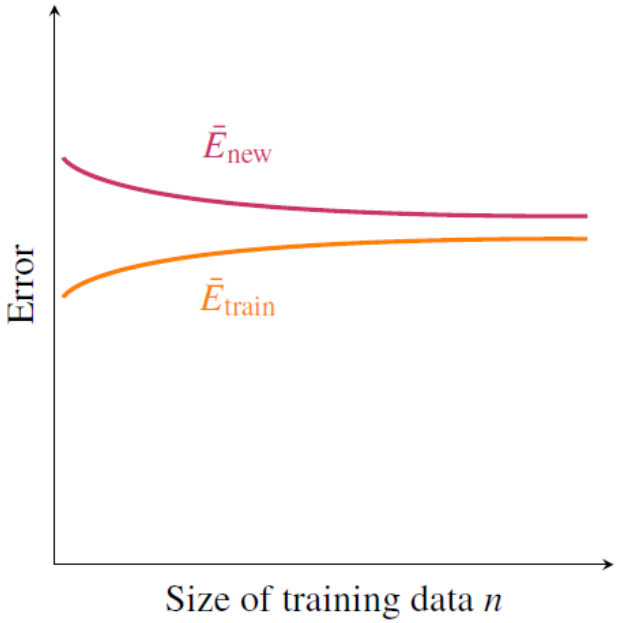
- Normally $generalization\ gap > 0$

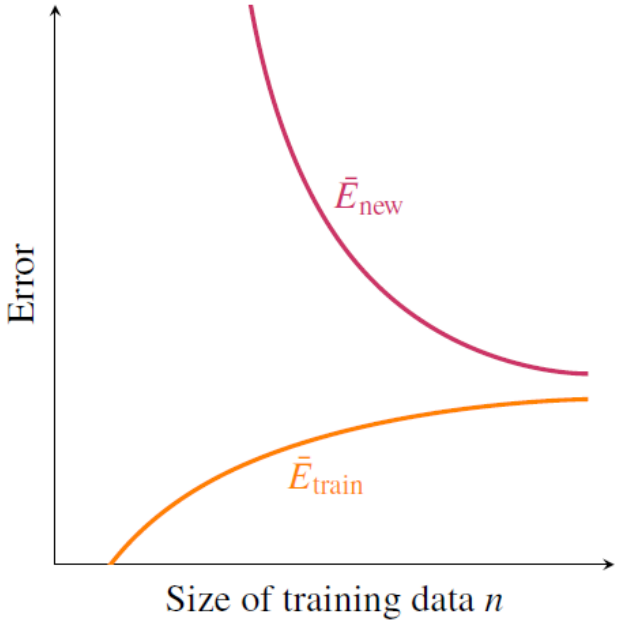
- How to estimate:
  1. Generate training and validation data from the generating process
  2. Estimate training and validation errors
  3. Repeat 1-2 m times
  4. Substract average train error from average validation error

- Note: Normally $\bar{E}_{new} > \bar{E}_{train}$ but this is not always the case for individual training and validation errors

Model selection

- Dependence of data size





Model selection: comments

- Choose model with smallest prediction error
  - If $E_{new} \approx E_{train}$ then model is probably too simple
  - If $E_{new} \gg E_{train}$ then then probably overfitting

- If the best training error too large→ change model or get more data…

- Increasing n may help a lot for complex models

Bias-variance tradeoff

- Bias of an estimator

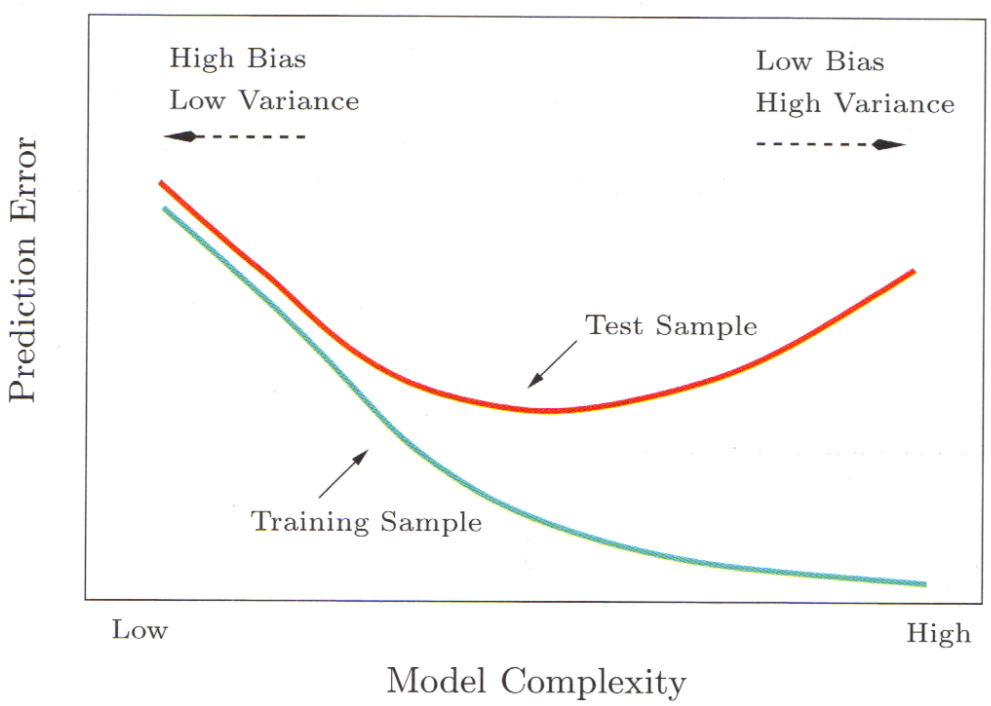
$$Bias(\hat{y}\,(x_0)) = E[\hat{y}(\boldsymbol{x}_*) - f(\boldsymbol{x}_*)]$$

  - $f(\boldsymbol{x}_*)$ is expected response, $y = f + \epsilon$,
  - $Var(\epsilon) = \sigma^2$
  - If $Bias(\hat{y}(\boldsymbol{x}_*)) = 0$, the estimator is **unbiased**
  - ML estimators are asymptotically unbiased if the model is enough complex
  - However, unbiasedness does not mean a good choice!

Bias-variance tradeoff

- Assume error function is $E(y, \hat{y}) = (y - \hat{y})^2$

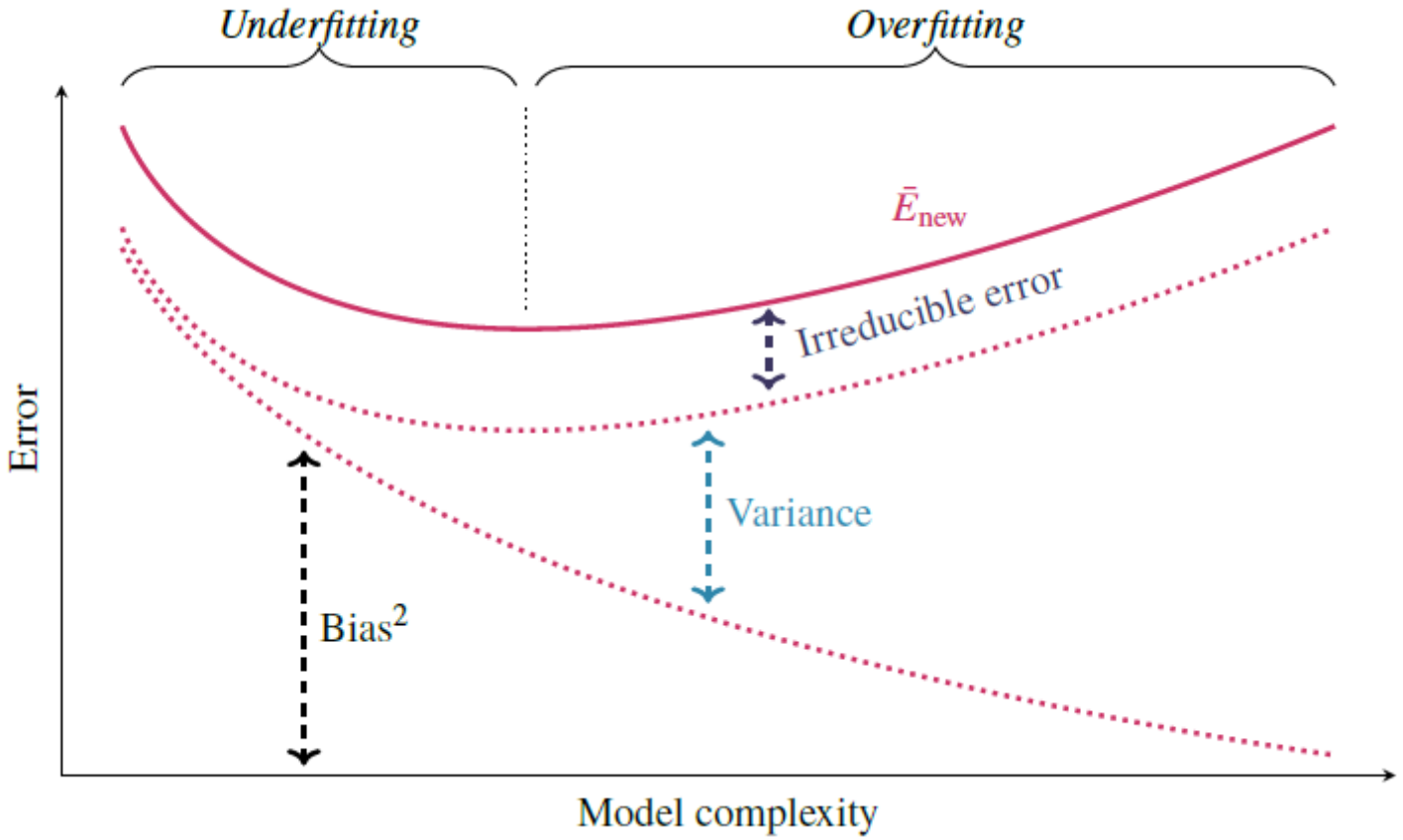
$$\bar{E}new\left(y(\boldsymbol{x}_*), \hat{y}(\boldsymbol{x}_*)\right) = \sigma^2 + Bias^2\left(\hat{y}(\boldsymbol{x}_*)\right) + Var(\hat{y}(\boldsymbol{x}_*))$$



When error is not quadratic, no such nice formula exist

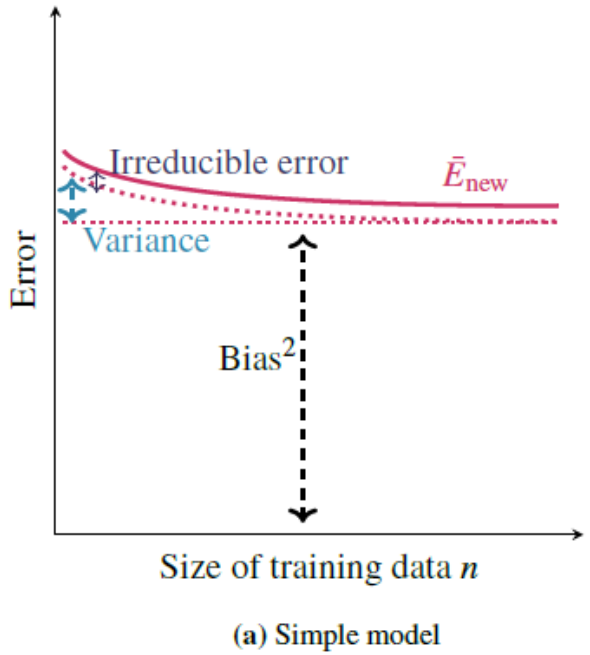
Bias-variance tradeoff

- Individual contributions

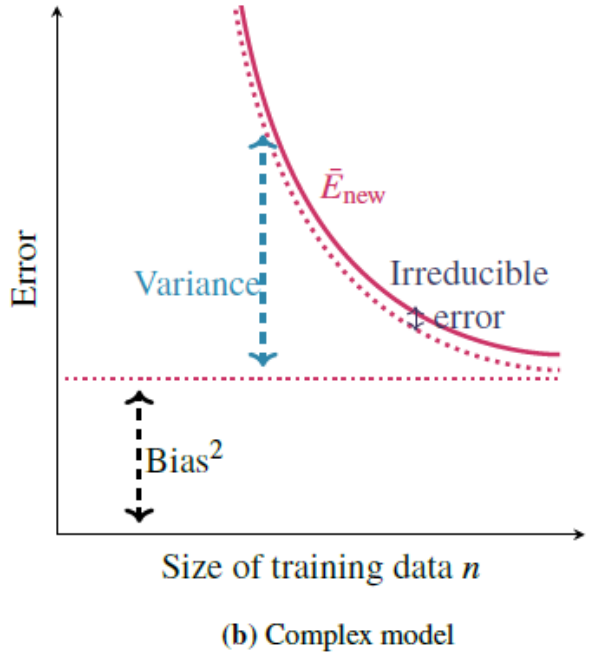


Bias-variance tradeoff

- Influence of n



(a) Simple model



(b) Complex model

Model selection

Example Computer Hardware Data Set : performance measured for various processors and also

- Cycle time
- Memory
- Channels
- …

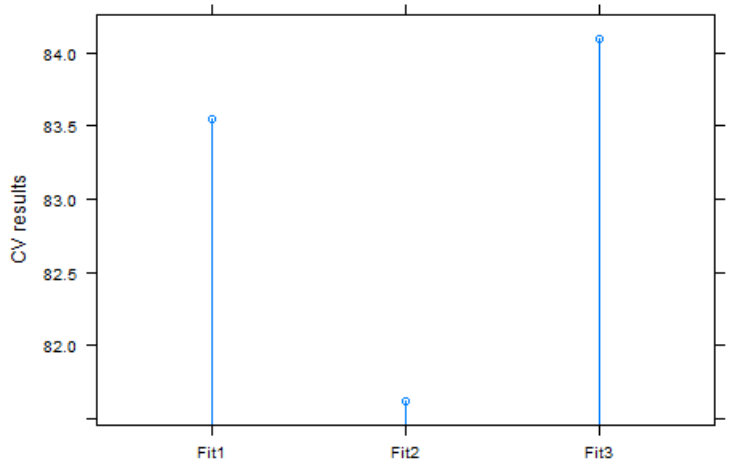
Build model predicting performance

Cross-validatation

- Try models with different predictor sets

```
data=read.csv("machine.csv", header=F)
library(cvTools)


fit1=lm(V9~V3+V4+V5+V6+V7+V8, data=data)
fit2=lm(V9~V3+V4+V5+V6+V7, data=data)
fit3=lm(V9~V3+V4+V5+V6, data=data)
f1=cvFit(fit1, y=data$V9, data=data,K=10,
foldType="consecutive")
f2=cvFit(fit2, y=data$V9, data=data,K=10,
foldType="consecutive")
f3=cvFit(fit3, y=data$V9, data=data,K=10,
foldType="consecutive")
res=cvSelect(f1,f2,f3)
plot(res)
```



Model evaluation

- Binary classification
- The choice of the thershold $\mathrm{p}(y = 1|x) > r \rightarrow classify$ "1" affects prediction→which classifier is better?
- **Confusion matrix**

		PREDICTED		
TRUE		0	1	Total
	0	TN	FP	N
	1	FN	TP	P

Model evaluation

- **Accuracy**

$$acc = \frac{TP + TN}{P + N}$$

- **True Positive Rates** (TPR) = **sensitivity** = **recall**
  - Probability of detection of positives: TPR=1 positives are correctly detected

$$TPR = TP/P$$

- **False Positive Rates** (FPR)
  - Probability of false alarm: system alarms (1) when nothing happens (true=0)

$$FPR = FP/N$$

- **Specificity**

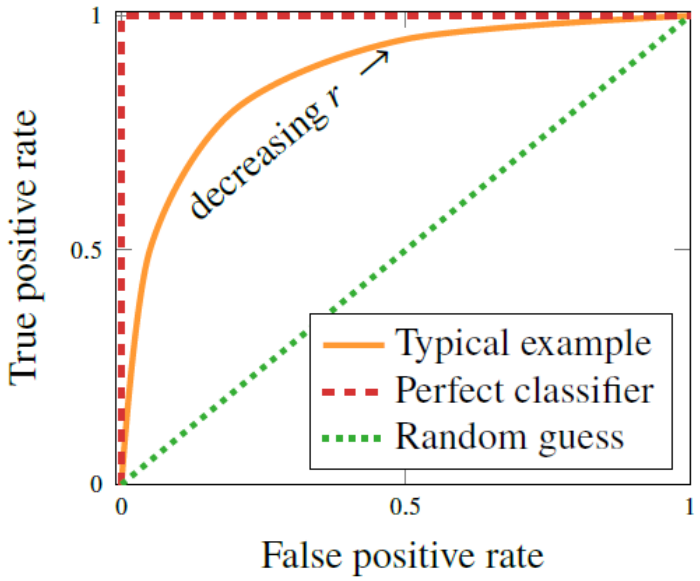
$$Specificity = 1 - FPR$$

- **Precision**

$$Precision = \frac{TP}{TP + FP}$$

ROC curves

- **ROC**=Receiver operating characteristics
- Use various thresholds, measure TPR and FPR
- Same FPR, higher TPR→ better classifier
- Best classifier = greatest Area Under Curve (**AUC**)



Imbalanced classes

- Note: if 1000 ”0” cases and 10 ”1” cases, classifying all to ”0” results in 99% accuracy

- **F1 score**: take into account class imbalance

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

- **FB score**: different costs (β importance of recall versus precision)

$$F_\beta = \frac{(1 + \beta^2) \cdot precision \cdot recall}{\beta^2 precision + recall}$$

Imbalanced classes: precision-recall curve

- Better than ROC for imbalanced classes