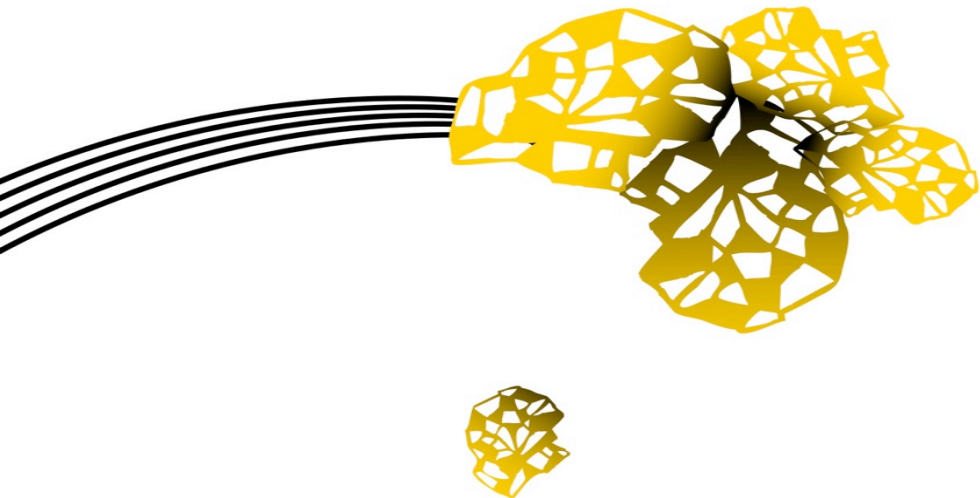


GENERALIZATION, OVERFITTING, VALIDATION, REGULARIZATION, PERFORMANCE MEASURES

MANNES POEL

GWENN ENGLEBIENNE



Announcements

- Homework assignment 3 is available on BB.
- Put your name and group number on the homework assignments.
- We will start grading HW 2 this week. Some intermediate results may appear on Canvas, so don't immediately contact us if your grade is not complete. Will announce when grading is ready.
- Written exam is multiple choice and open book. Example exam will be available in due time.

Questions about


- Homework assignments
- Lab sessions

Homework assignments & Lab session

- Summarizing weight updates:

$$w^{new} = w^{old} + \eta * dw$$

$$dw = -\nabla_w E$$

$$\nabla_w E = \sum \left(\frac{\partial E}{\partial y_n} \right) \times \left(\nabla_w y_n \right)$$


- First term depends on Error function, second term on ML model!

Batch versus stochastic

Batch learning (update w after each batch of training samples).

- Stochastic (update w after each training samples).
- Order of training set relevant?
- In stochastic gradient descent randomize training set in each round/episode/epoch.

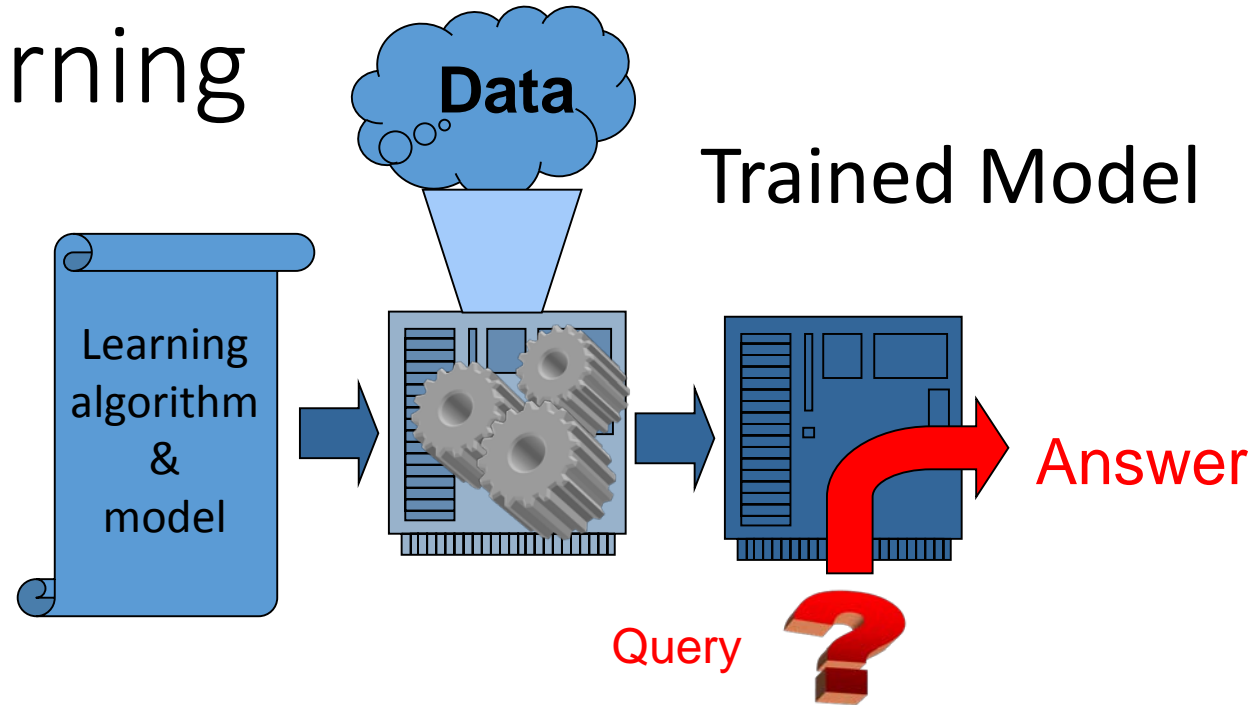
Topics of today

- Performance measures
- Generalization
- Overfitting
- Validation
- Regularization

Not necessarily in this order: it will be a mix or cocktail of all.

What do we want to achieve with Machine Learning

Machine Learning



The starting point is that one wants to design and learn the **best** ML model for the task at hand.

In this context **best** means the model with the highest **performance**.

Performance measures

Error function and classification


- During training the learning algorithm tries to minimize an error function. For instance:

$$E_w(f) = - \sum_{n=1}^N t_n \log(f_w(x_n)) + (1 - t_n) \log(1 - f_w(x_n))$$

- But there is no clear relation between the classification performance and the error. Of course in general, the lower the error the better the classification performance (accuracy of the classifier). But a small reduction in the error function does not imply an improvement in the classification performance.

Classification Performance: Confusion matrix

- A confusion matrix is a multidimensional measure of performance from which a lot of relevant information can be deduced.
- Example for a two class problem ham/spam:

Act /Pred	Spam	Ham	 Predicted class
Spam	97 (TP)	51 (FN)	
Ham	11 (FP)	825 (TN)	

Accuracy, precision and recall

- Accuracy?
- $(825+97)/(825+11+51+97)$
- Precision for spam?
- $97/(97+11)$
- Recall for ham?
- $825/(825+11)$

Act /Pred	Spam	Ham
Spam	97	51
Ham	11	825

Also sensitivity and specificity can be computed from this matrix

Trade off between Precision and Recall

- Assume that we trained a logistic classifier/linear discriminant σ such that $\sigma(x) = P(\text{spam}|x)$.
- Classification rule: $\sigma(x) > \theta$ then email x is classified as spam.
- $\theta = 0$: All emails are classified as spam.
- Precision and recall for spam if $\theta = 0$?
Recall = 1 but Precision depends on number of samples.
But TP_rate = FP_rate = 1
- $\theta = 1$: All emails are classified as Ham (non-spam). Precision = 1 and Recall = 0.
TP_rate = FP_rate = 0.
- So there is a trade off between precision and recall (true positive rate and false positive rate) depending on θ . Can be captured in the **receiver operating characteristic (ROC)** curve for a certain class.

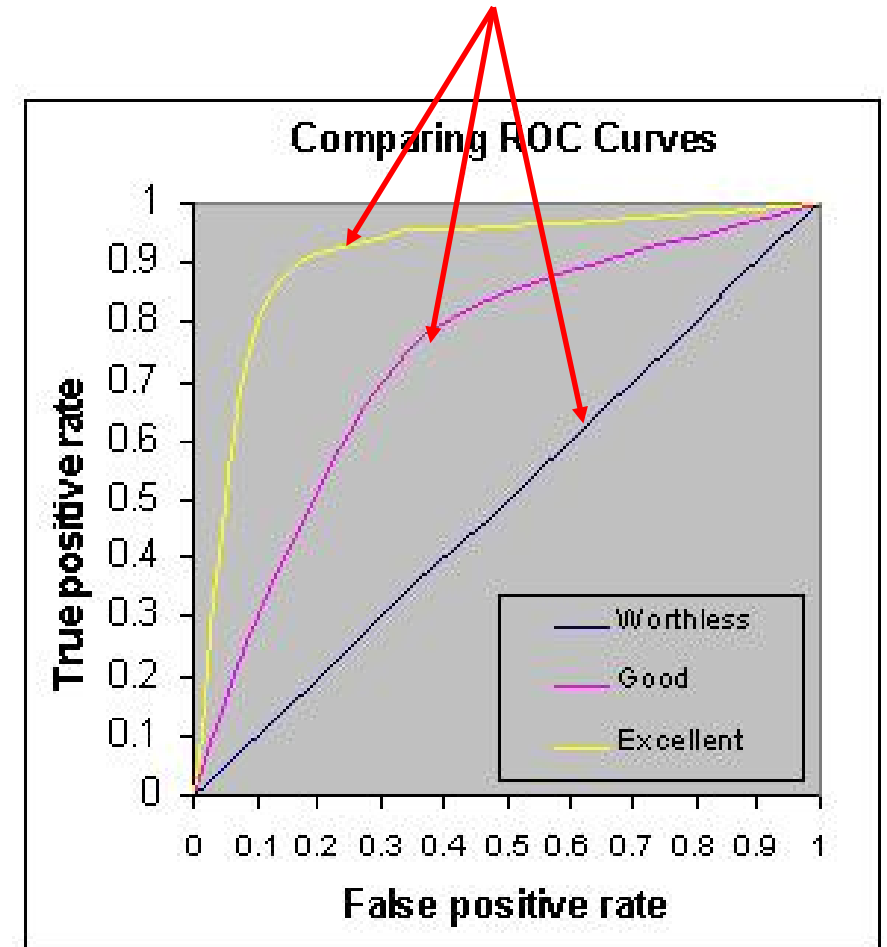
$\theta = 0$:

Act /Pred	Spam	Ham
Spam	148	0
Ham	836	0

ROC curve

If one varies θ one gets curves like this, depending on your model

- Vertical axis TP rate and horizontal axis FP rate.
- $\theta = 0$: TP=1 and FP=1
- $\theta = 1$: TP=0 and FP=0
- Area under Curve (AUC) is the area under the ROC curve. This is also an often used performance measure for comparing classifiers.



Reflection on Learning

- Given a classification problem: data set d and labels l .
- $X = \{(x_n, t_n) | n = 1 \dots N\}$
- Model: logistic classifier determined by weights w .
 - If $\sigma(w^T x) > \frac{1}{2}$ then x has classification label 1 else x has label 0.
 - Defined error function

$$E(w) = - \sum_{n=1}^N t_n \log(\sigma(w^T x_n)) + (1 - t_n) \log(1 - \sigma(w^T x_n))$$

$$y_n = \sigma(w^T x_n)$$

- Use gradient descent to determine w^* which minimizes $E(w)$
- This w^* will be used in the classification.

More general

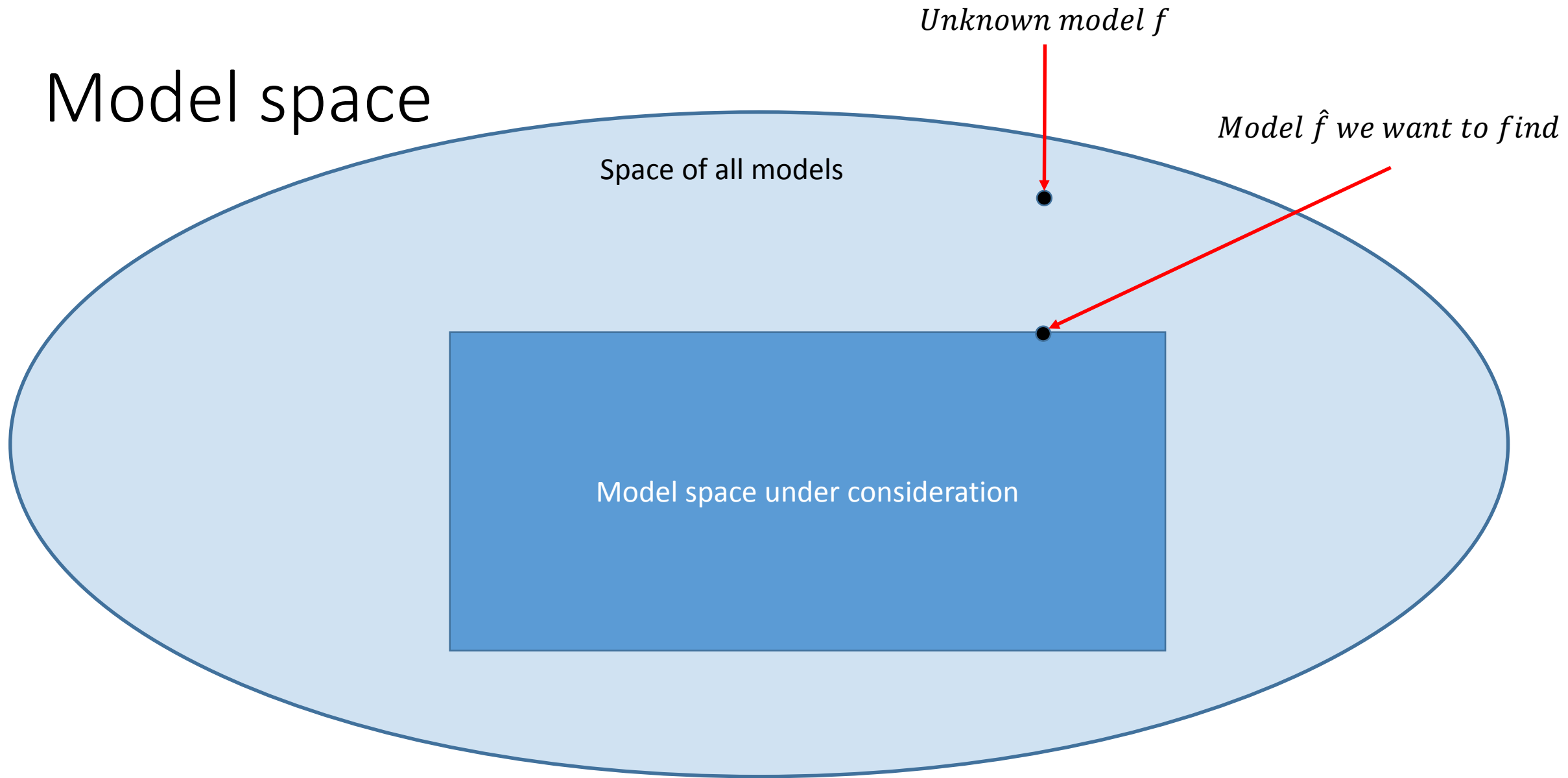
- Given the data set and labels we want to find a model (or function) f such that the error function

$$E(f) = - \sum_{n=1}^N t_n \log(f(x_n)) + (1 - t_n) \log(1 - f(x_n))$$

is minimal.

- Need a hypothesis space or model space from which we draw f . Otherwise problem is unsolvable.

Model space



How is \hat{f} defined

- \hat{f} should be the model in our model space which is *closest* to f .
- How to define the distance between \hat{f} and f ? Problem is that we don't know f . Otherwise we could maybe apply some clever mathematical tricks to find \hat{f} .
- But we have some example data, most of the time noisy, of the behavior of f .
- This can be used to define an Error function E and the \hat{f} we are looking for should minimize this E .

How to find \hat{f} ?

- Exhaustive search: not feasible. Model space under consideration (hypothesis space) is infinite.
- Clever search such as Genetic Algorithms.
- Parametrized model space, parameter is w .
 - Start with initial guess and update w for instance by gradient descent.
- No guarantee that one will find \hat{f} , let alone the unknown model or function f one is looking for.

MP approach

- Define f_{MP} as follows:
 - For input x determine index i of x in data set d . Return $l[i]$.
- In other words a look-up table $f_{MP}(x_n) = t_n$
- Then

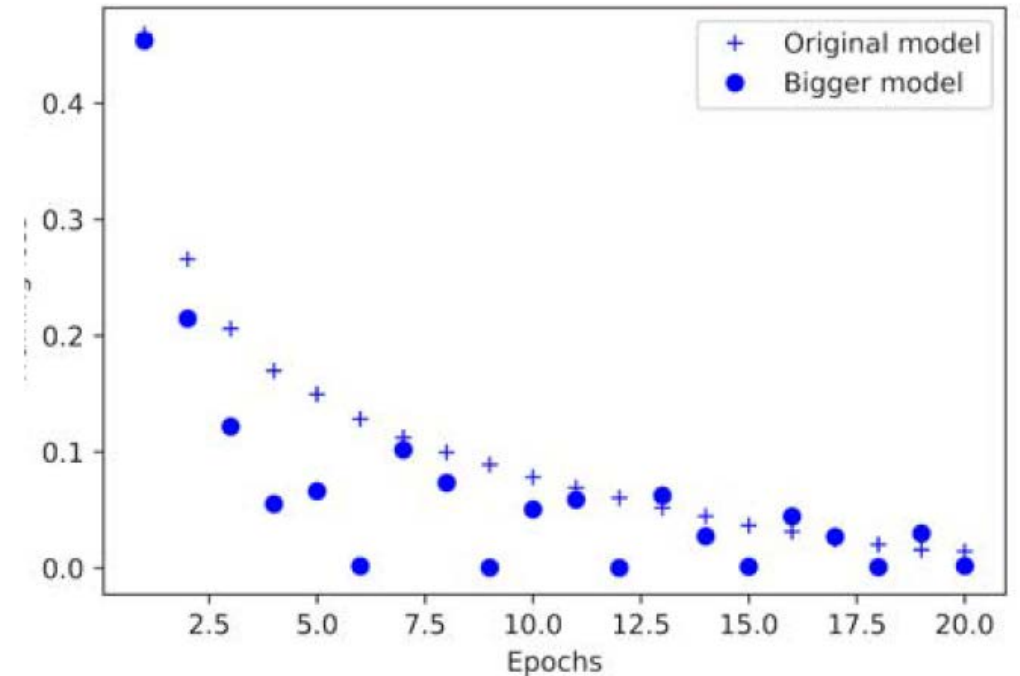
$$\begin{aligned} E(f_{MP}) &= - \sum_{n=1}^N t_n \log(f_{MP}(x_n)) + (1 - t_n) \log(1 - f_{MP}(x_n)) = \\ &= - \sum_{n=1}^N t_n \log(t_n) + (1 - t_n) \log(1 - t_n) = 0 \text{ because } t_n \in \{0,1\} \end{aligned}$$

So what is the problem?

Overfitting

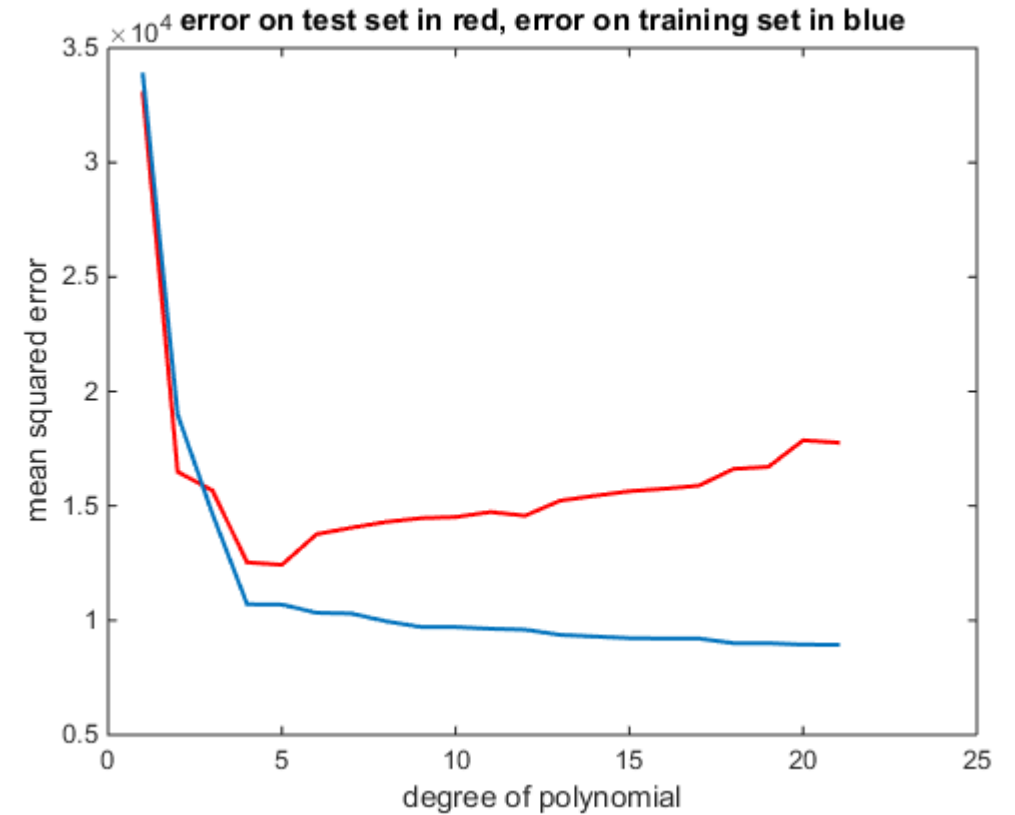
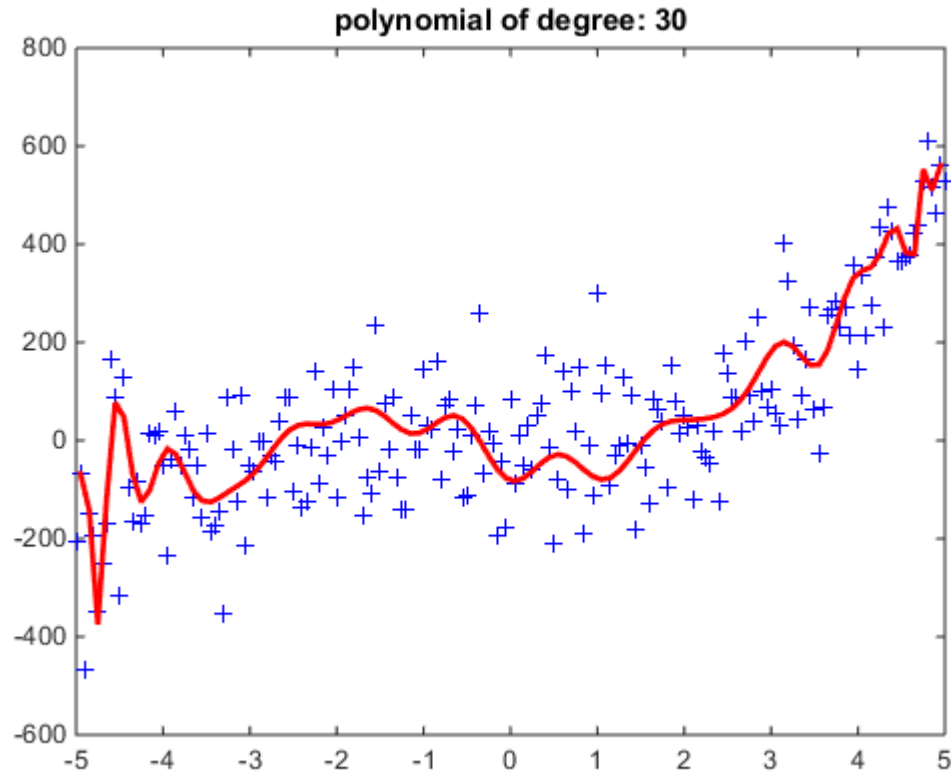
- Too much focus on training data.
 - The more complex the model the better it will fit the training data.
- Example: Most clear way to show this is in a regression case. But the same holds for classification.
- See Matlab (☺) simulation.

Loss on training set



From Chollet: Deep Learning with Python

Summary polynomial fitting

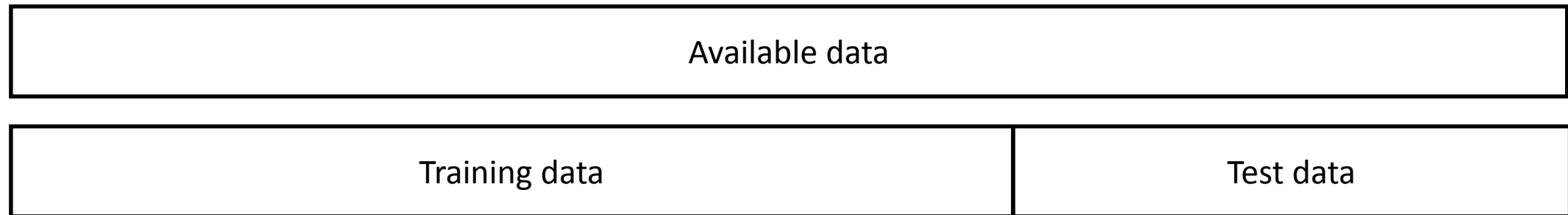


Generalization

- What is the problem of MP solution f_{MP} :
 - Works perfect for the training data but only defined for input which is in training data.
 - No solution for classifying new (not in training data, unseen) data points.
- What we want is a model f which performs *optimal* on new, unseen, data. In other words when model f is used in a real-life application.
- Needed before using f : Assessment of performance of model f on new, unseen, data. This is called the *generalization performance*.
- So we need to model unseen data (real-life application), the future.

Generalization performance

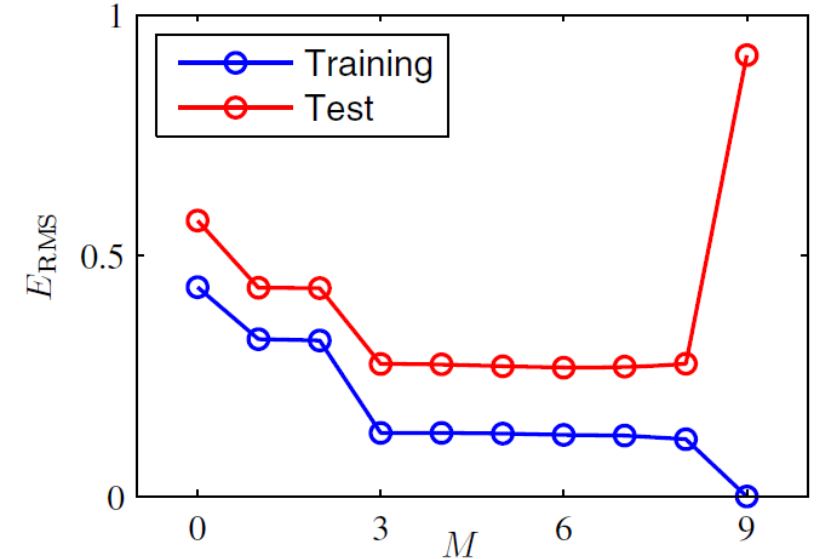
- Generalization performance is an estimate of the performance of the model on new unseen data.
- How to estimate this performance:



- Test data is used to estimate generalization performance. Models the future application of the model.
- *So the test data should be a good representation of the future application of the model!*

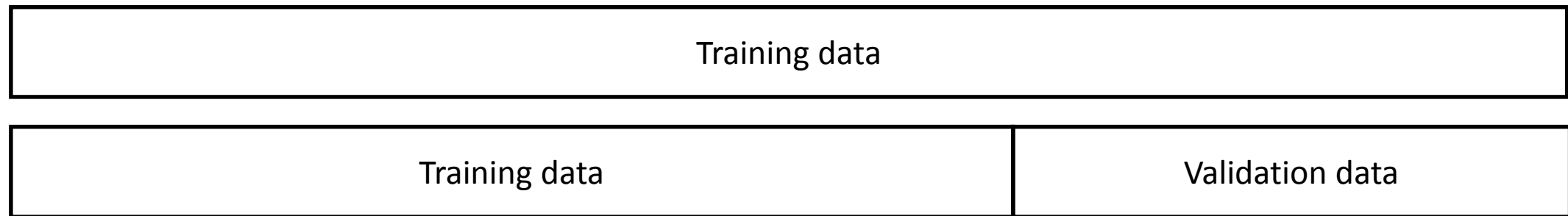
Modeling unseen data

- Split data in training and test set:
- Select model, for instance logistic classification
 - Train model (w) on training set and compute performance or error of model on test set.
 - Typical behavior:
 - Stop when error on test set increases.
- Or train a lot of different models and test each model on the test. Select model with highest performance on the test set.
- Is this a **sound methodology**?
- **NO: can result in overfitting on this particular test set**, resulting modeling the noise and not the underlying model.



Generalization performance

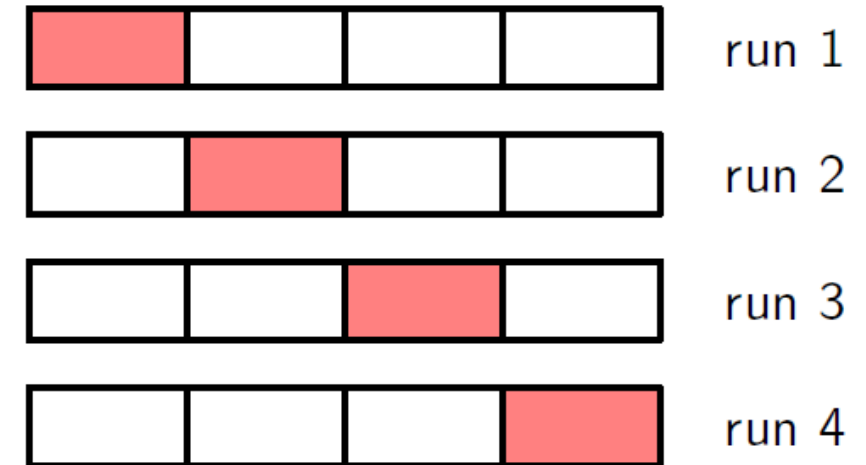
- Validation set is used for model selection and stopping learning. How to estimate this performance.



- But performance of model can depend on split (bad versus good split)

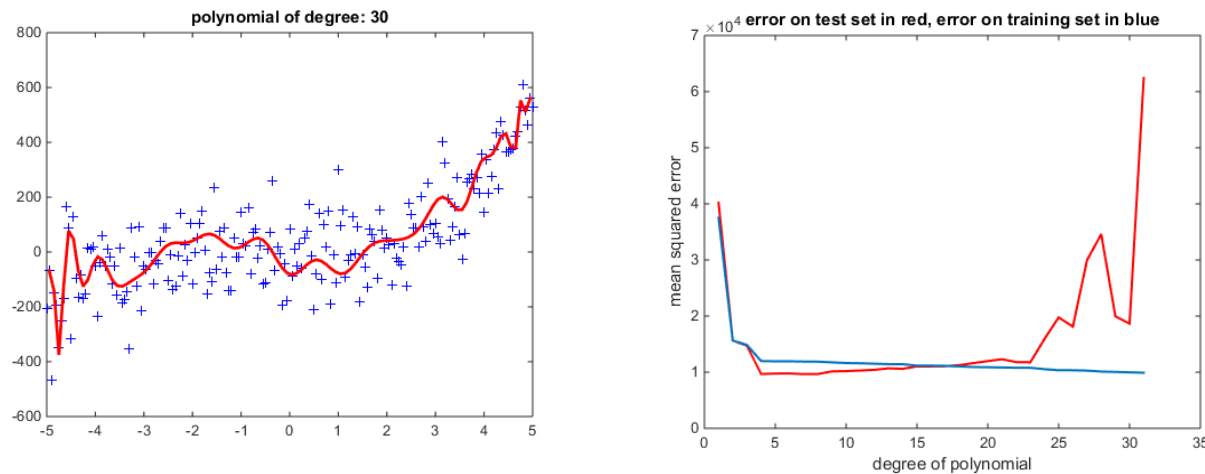
k-fold cross validation (k-CV)

- Split training set in k pieces:
- Train models on $(k-1)$ pieces and evaluate on remaining piece.
For instance models of degree 1 up to n or in general of different form or different parameter settings.
- k evaluations to estimate performance for instance mean and variance.
- Use these to select best model
- Train model on complete training set and estimate performance on test set.



Regularization

- Let's look back at the overfitting example.



- Some coefficients of the polynomial:
 $10^3 \times [-0.3823 \quad 0.7583 \quad 0.8246 \quad -1.2305 \quad -0.9694 \quad 1.0494 \quad 0.4699]$
- Typical phenomena of overfitting: **large weights!**

Regularization

- Prevent large weights by adding penalty term to error function:

$$E = E_D + \lambda E_w$$

- E_D is error on training set
- E_w is error term for large weights:
 - $E_w = 1/2 \sum_{i=1}^n w_i^2$ (observe that index $i=0$ is not included)
- Consequence for $i>0$:

$$\nabla_{w_i} E = \nabla_{w_i} E_D + \lambda w_i$$

Regularization: result

$$\mathbf{w}^{new} = \mathbf{w}^{old} + d\mathbf{w}$$

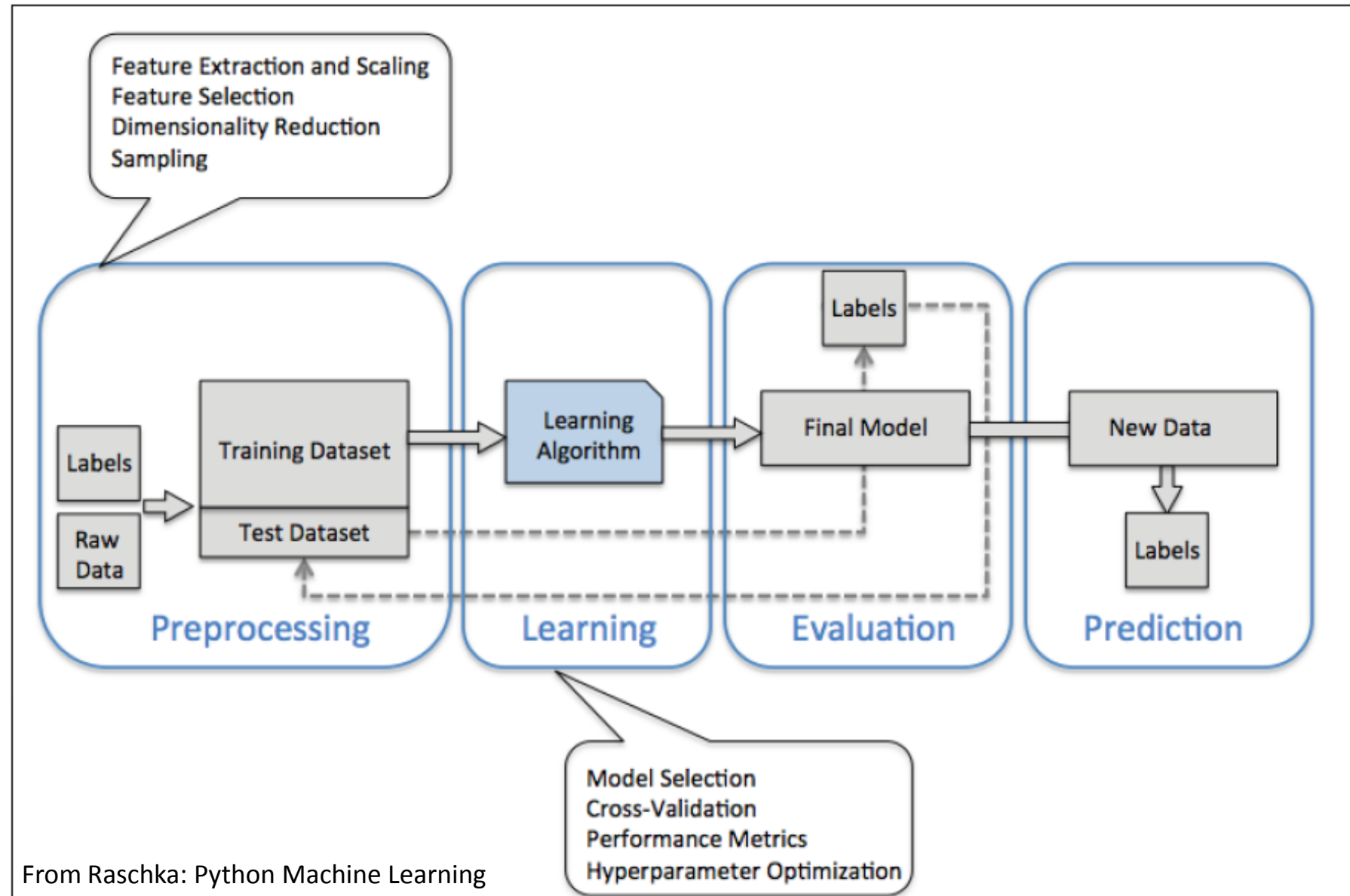
$$d\mathbf{w} = -\eta \nabla_{\mathbf{w}} E = -\eta \nabla_{\mathbf{w}} E_D - \eta \lambda \mathbf{w}$$

- Usually one replaces $\eta\lambda$ by λ .
- In case of logistic discrimination for $i > 0$:

$$dw_i = \eta(\text{target} - \text{output})x_i - \lambda w_i$$

- But this also induces another problem: How to determine λ ?

ML pipeline



What did we learn

- Basic knowledge of:
 - Overfitting & Generalization
 - Validation & Testing
 - Regularization
 - Performance measures:
 - Confusion matrix
 - Accuracy
 - Precision
 - Recall
 - ROC curve & AUC
 - ML Pipeline

Reading material

- Slides
- Chapter 1
- Notes, see Week 1.

