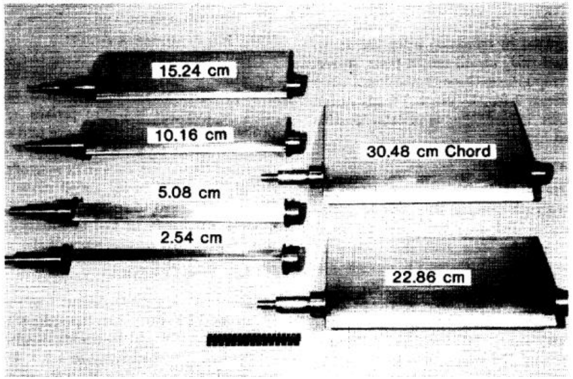
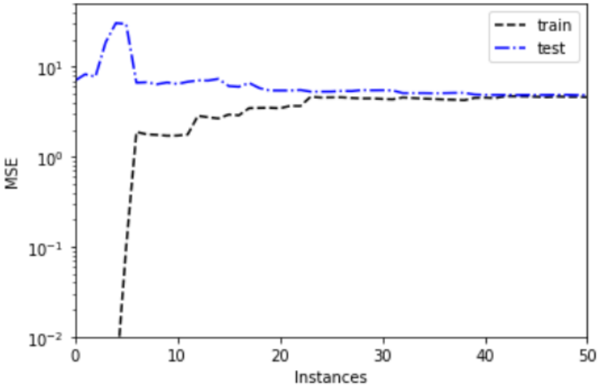
Karlsruhe Institute of Technology

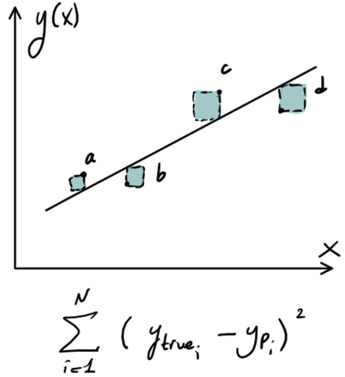
Data Driven Engineering I: Machine Learning for Dynamical Systems

Analysis of Static Datasets I: Regression

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer







One page summary: Intro. to ML

* There are 4 main learning strategies, mainly based on feedback info.
    * Error-based, similarity-based, info-based, probab.-based

* The goals of 4 main ML tasks is very relevant to our learning strategies

* ML := ill-posed problem ≫ There will be many solutions for a problem.

* Nature & quality of data affects outcomes drastically !

* ML is very similar to cooking: Follow the proposed steps for a generic project.

Today's Agenda

Karlsruhe Institute of Technology

Basic Steps to Follow:

0.) Understand the business/task.
1.) Understand the data.
2.) Explore & prepare the data.
3.) Shortlist candidate models.
4.) Training the model
5.) Evaluate the model predictions.
6.) "Serve" the model!

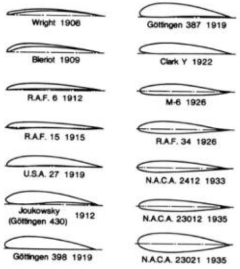
Regression

#0 Understanding the task

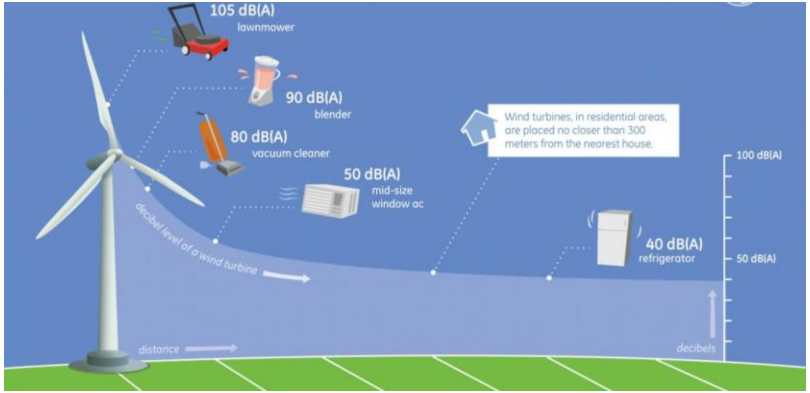
- **Problem**: NACA 0012 Airfoil Noise Prediction based on Wind Tunnel Testing

- **Noise** generated by an aircraft is an **economic** (efficiency) and **enviromental** issue.

- One component of the noise the **self-noise of the airfoil**: interaction of the airfoil with its own boundary layer



1917, the NACA Technical Report No. 18 titled "Aerofoils and Aerofoil Structural Combinations," was released.



#0 Understanding the task

Karlsruhe Institute of Technology

- ❑ Engineering: semi-emprical models (Brooks)
- ❑ Five self-noise mechanisms due to specific boundary-layer phenomena have been identified
- ❑ The database is from seven NACA0012 airfoil blade sections of different sizes tested at wind tunnel speeds up to Mach 0.21 and at angles of attack from 0°to 25.2°.
  - ✓ Freq. of noise
  - ✓ Angle of attack
  - ✓ Free stream velocity
  - ✓ Geometry of the airfoil

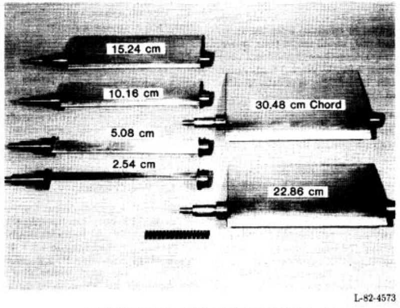
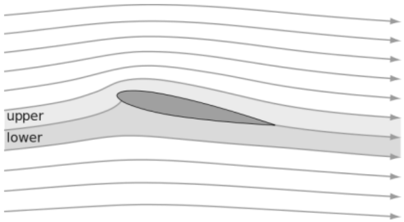
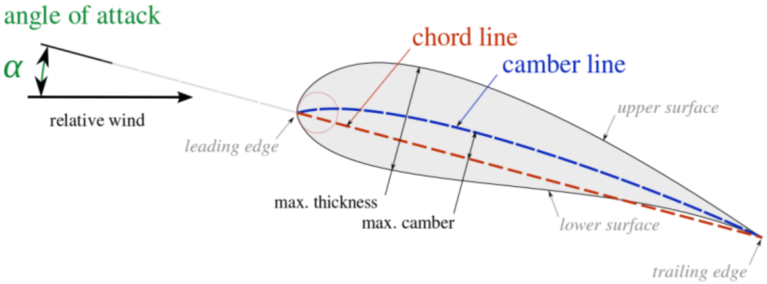


Figure 2. Two-dimensional NACA 0012 airfoil blade models.





#1 Understanding the data

- ❑ Check the data source: understand what the data refers to
- ❑ Objective: understand the characteristics of the data
- ❑ Look at the feature columns:
  - ❑ Any missing values?
  - ❑ Any features with NaN values?
  - ❑ Uniqueness of the dataset? (“cardinality”)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1503 entries, 0 to 1502
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   frequency               1503 non-null   int64
 1   angle_attack            1503 non-null   float64
 2   chord_length            1503 non-null   float64
 3   Free-stream_velocity    1503 non-null   float64
 4   displacement_thickness  1503 non-null   float64
 5   sound_pressure          1503 non-null   float64
dtypes: float64(5), int64(1)
memory usage: 70.6 KB
```

```
data.head(5)
```

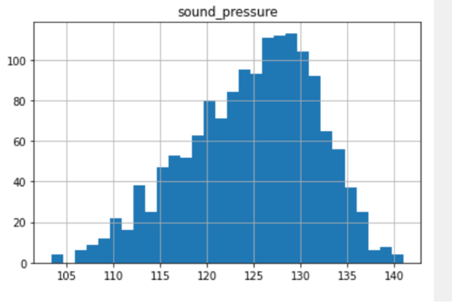
	frequency	angle_attack	chord_length	Free-stream_velocity	displacement_thickness
0	800	0.0	0.3048	71.3	0.002663
1	1000	0.0	0.3048	71.3	0.002663
2	1250	0.0	0.3048	71.3	0.002663
3	1600	0.0	0.3048	71.3	0.002663
4	2000	0.0	0.3048	71.3	0.002663

colab

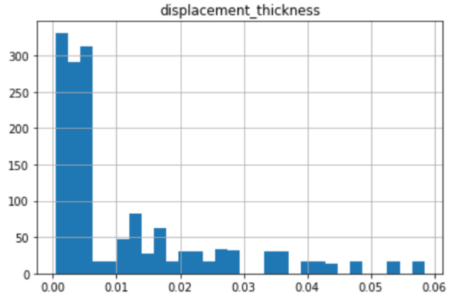
#2 Exploring the data

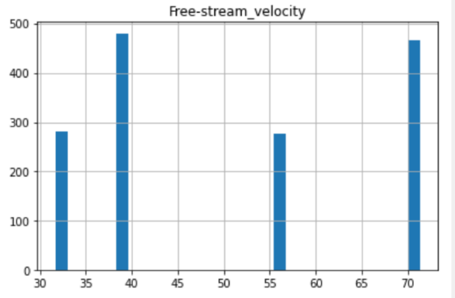
- ❑ **Objective**: generate a data quality report

- ❑ Using standard statistical measures of central tendency and variation
  - ❑ tabular data and visual plots
  - ❑ mean, mode, and median
  - ❑ standard deviation and percentiles
  - ❑ bars, histograms, box and violin plots



- ✓ Missing values,
- ✓ Irregular cardinality problems,
  - ▪ 1 or comparably small
- ✓ Outliers
  - ▪ invalid outliers and valid outliers





#2 Exporing the data: Correlation Matrix

- Shows the correlation between each pair of features

$$\underbrace{Cov(a,b)}_{\text{Features}} = \frac{1}{n-1}\sum_{\underbrace{i=1}_{\text{instance}}}^{n}\left[(a_i-\underbrace{\bar{a}}_{\text{mean}}) \times (b_i-\underbrace{\bar{b}}_{\text{mean}})\right]$$

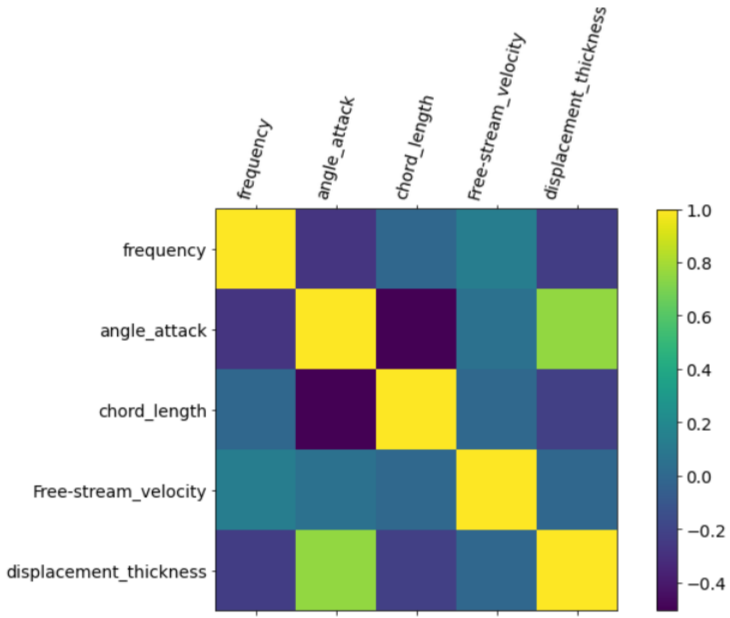
- Normalized form of “covariance”

$$\Downarrow$$

$$Corr(a,b) = \frac{Cov(a,b)}{SD(a) \times SD(b)}$$

* Normalized
* Dimensionless
Easy to interpret

- Ranges between −1 and +1



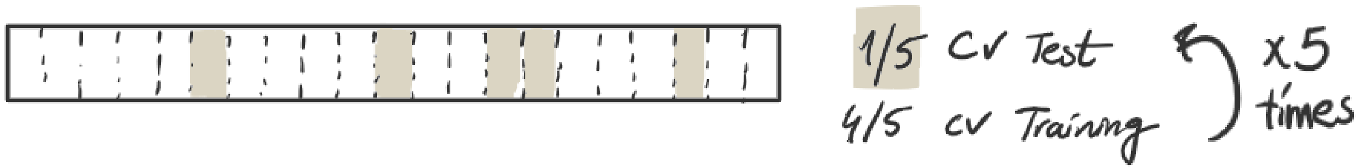
colab

#2 Preparing the Data

- Classification >> supervised >> **training & test split**

- Reducing overfitting via **cross-validation**: take **random portions** of the data to build a model

- **k-fold** method: k = 5; (typically 10)

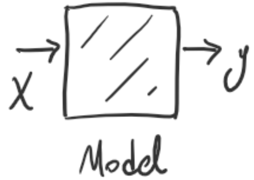


colab

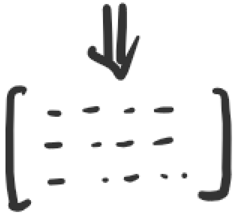
Model Selection: Linear Regression 1

```
data.head(5)
```

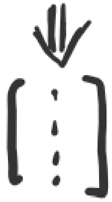
	frequency	angle_attack	chord_length	Free-stream_velocity	displacement_thickness
0	800	0.0	0.3048	71.3	0.002663
1	1000	0.0	0.3048	71.3	0.002663
2	1250	0.0	0.3048	71.3	0.002663
3	1600	0.0	0.3048	71.3	0.002663
4	2000	0.0	0.3048	71.3	0.002663



$$\underbrace{\text{Model}}_{y \approx f(x, w)}$$



features X — What we know



labels y — What we want

① $y_p = w_0 + w_1 x_1 + w_2 x_2 + \dots w_n x_n$ } linear func. w_i & x_i

(bias)

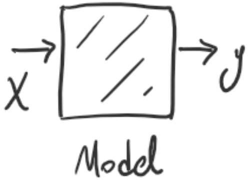
② $y_{True_i} = y_{p_i} + Error_i$ } Error Metric (norm) := Goodness of a fit

③ Extended to Nonlinearity via ϕ

$y_p = w_0 + \sum w_i \, \phi(x_i)$ → Basis functions → x^i (polynomial), $\sigma\left(\frac{x-\mu}{s}\right)$ (sigmoidal), ...

linear   nonlinear

Model Selection: Linear Regression 2



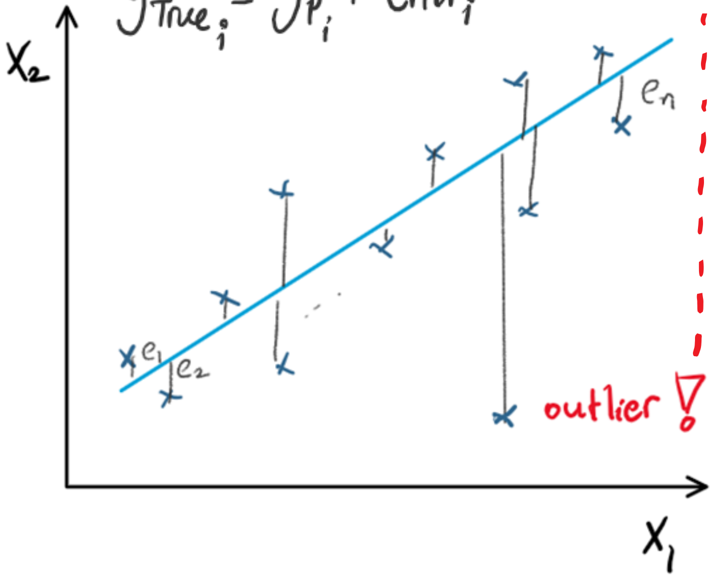
② $y_{True_i} = y_{p_i} + Error_i$ } Error Metric (norm) $:=$ Goodness of a fit

- Maximum error (ℓ_∞) $\max\limits_{1 \le i \le n} |y_{true_i} - y_{p_i}|$
- Mean absolute error (ℓ_1) $1/n \sum\limits_{i=1}^{n} |y_{true_i} - y_{p_i}|$
- Least Squares error (ℓ_2) $\left(1/n \sum\limits_{i=1}^{n} |y_{true_i} - y_{p_i}|^2\right)^{1/2}$

Model Selection: Linear Regression 2

Error-based learning:

$y_{True_i} = y_{P_i} + Error_i$



Presence of outliers / limited observ.

$$\max_{1 \le i \le n} |y_{true_i} - y_{P_i}|$$

$$1/n \sum_{i=1}^{n} |y_{true_i} - y_{P_i}|$$

$$\left(1/n \sum_{i=1}^{n} |y_{true_i} - y_{P_i}|^2\right)^{1/2}$$

Errors will be dictated by outliers !

Regularization

Model Selection: Linear Regression 3

(4) Regularized Linear Regression

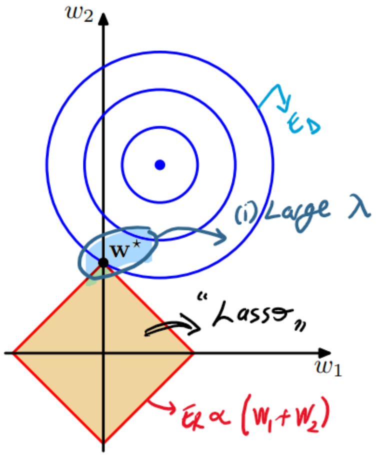
~~Obj~~ ~~"Over-fitted"~~ Regularized! $\Rightarrow$ $E_T = \underbrace{E_D}_{\text{Data error}} + \underbrace{E_R}_{\text{Regularization Error}}$

$$E_R \leftarrow \frac{\lambda}{2}\sum_{i}^{M}|w_i|^q$$

(a) Ridge Regression $\Rightarrow$ $E_R = \dfrac{\alpha}{2}\sum_{i=1}^{n} w_i^2$ (no bias here)

(b) Lasso $\Rightarrow$ $E_R \Leftrightarrow \ell_1$ norm $\quad E_R = \dfrac{\alpha}{2}\sum_{i=1}^{n}|w_i|$ (α is large; w is sparse!)

(c) Elastic Net $\Rightarrow$ $E_R = \overbrace{\frac{1-r}{2}\alpha\sum w_i^2}^{(1-r)\text{ Ridge}} + \overbrace{\frac{r\alpha}{2}\sum|w_i|}^{(r)\text{ Lasso}}$

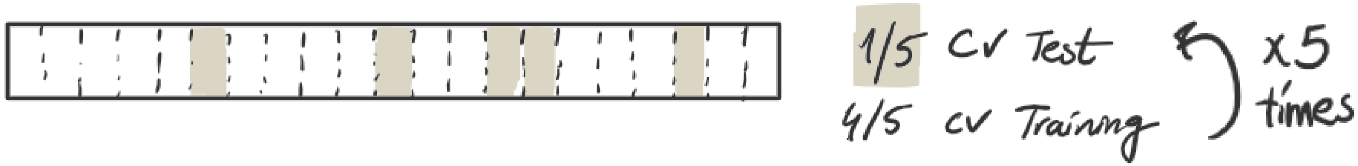


#4 Training the model

- Classification >> supervised >> **training & test split**

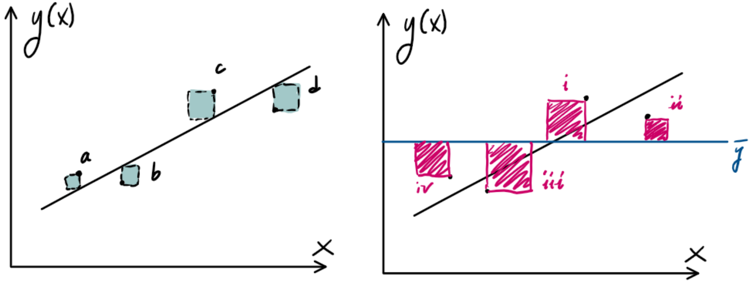
- Reducing overfitting via **cross-validation**: take **random portions** of the data to build a model

- **k-fold** method: k = 5; (typically 10)



#5 Evaluation of the results

- Coefficient of determination, R^2
  - Indicates the goodness of fit
  - Measure of generalization capability
  - Best possible score is 1.0
  - It can be negative



$$\sum_{i=1}^{N} \left(y_{true_i} - y_{P_i} \right)^2 \qquad \sum_{i=1}^{N} \left(y_{true_i} - \bar{y} \right)^2$$

$$R^2 \left(y_{true}, y_p \right) = 1 - \frac{\sum_{i=1}^{N} \left(y_{true_i} - y_{P_i} \right)^2}{\sum_{i=1}^{N} \left(y_{true_i} - \bar{y} \right)^2}$$

$$\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_{true_i}$$

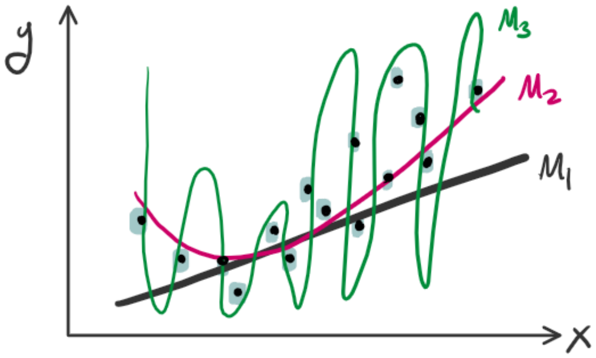
Karlsruhe Institute of Technology


Ateliers & Saveurs in Montreal

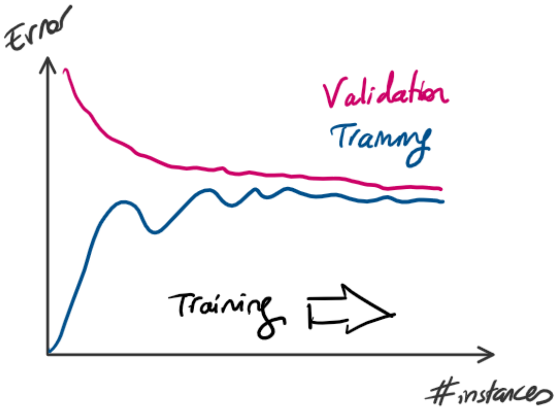
colab

#5 Evaluation of the results: Learning Curves

Case: 'Curve Fitting'



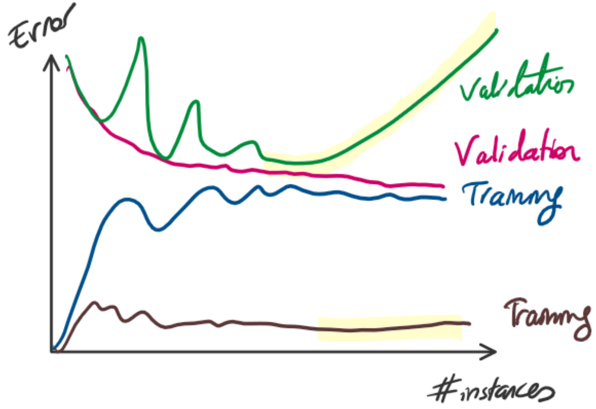
- ☐ Linear
- ☐ Polynomial (n=2)
- ☐ Polynomial (n=7)



(i) model learns

(ii) as it learns, model parameters generalizes.

(iii) E_D is found.

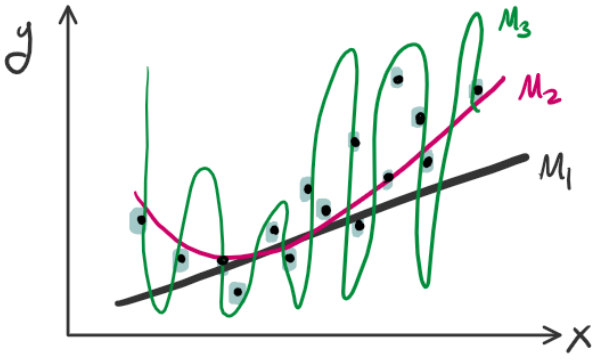


(i) Compare it with n=7;

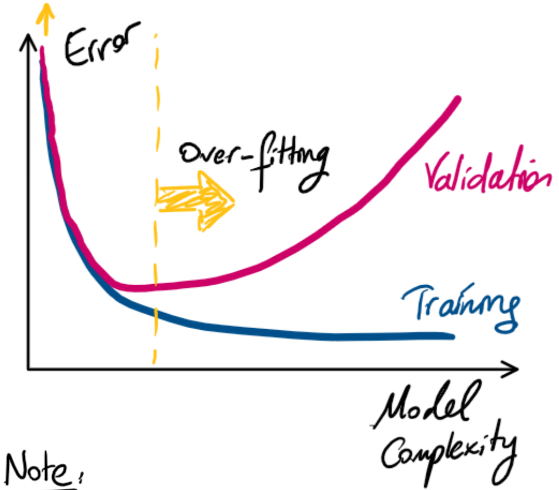
(ii) Divergence of E_D ⇒ Overfitting

#5 Evaluation of the results: Learning Curves

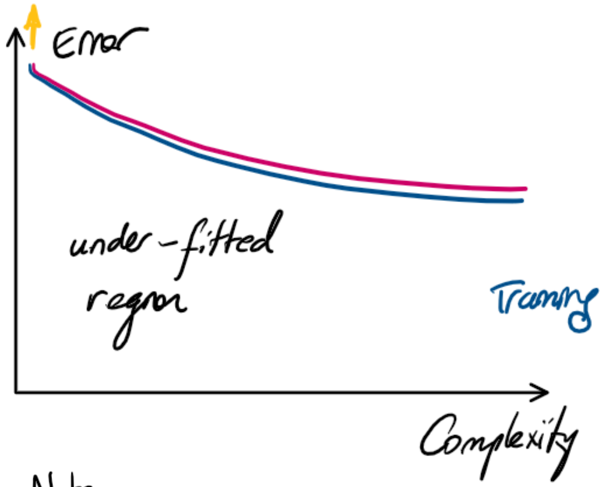
Case: 'Curve Fitting'



- ☐ Linear
- ☐ Polynomial (n=2)
- ☐ Polynomial (n=7)



Note:
number of instances are "large" enough

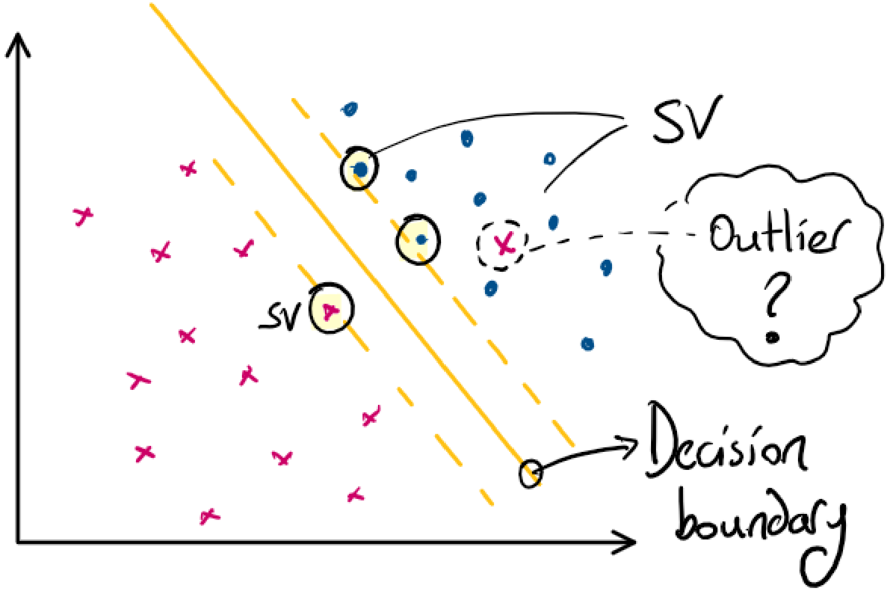


Note:
number of instances are "large" enough

colab

Model Selection: SVM for Regression 1

– SVM –

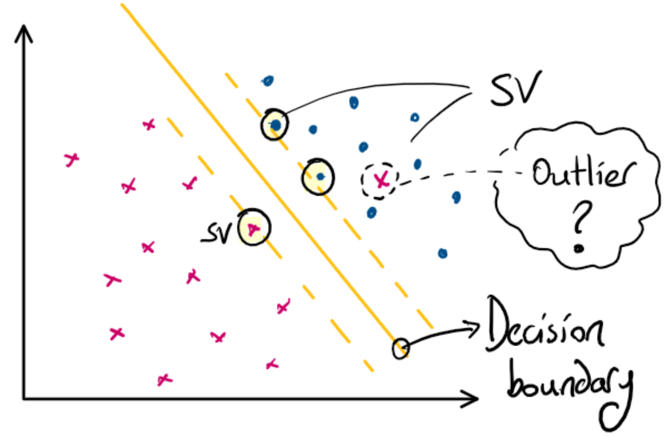


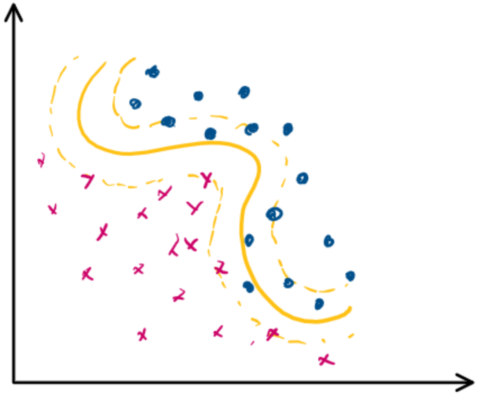
Classification

* fits a "street" between classes
* uses support vectors (SV)
* Decision is based on SVs, not other instances.
* Feature scaling is important
* outliers ⇒ "Soft Margin" (~C)
  ✓ limit margin violations
* must be linearly separable

Model Selection: SVM for Regression 2

— SVM —





Classification

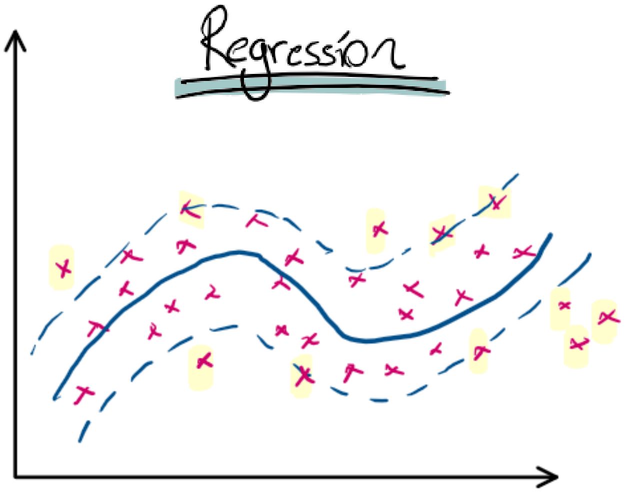
* linear decision bound. ⇒ X
* "Kernel Trick" $:= \phi(x)$

  (✓) introduce non-linearity

  (✓) "feature eng." without adding new features.

Model Selection: SVM for Regression 3

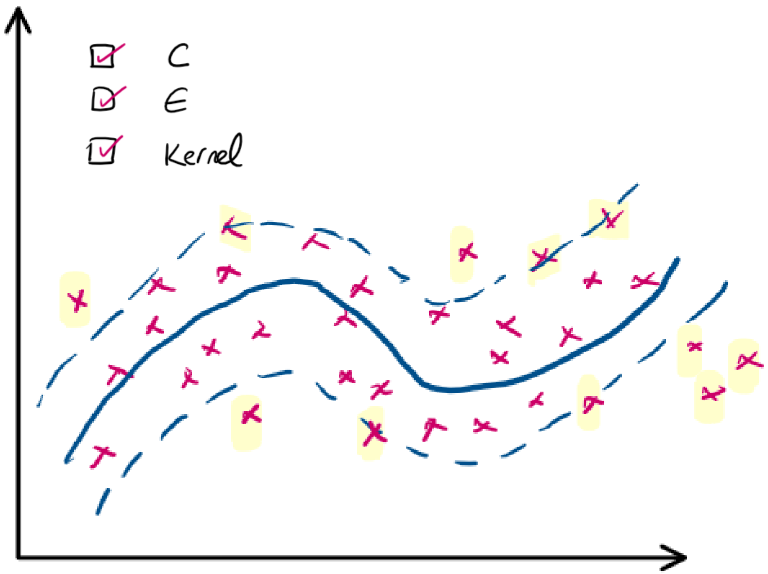
– SVM –



- Fit as many instance as possible
- "Street" width is controlled by margin ϵ.
- Convex optimization problem:
  - ☑ C
  - ☑ ϵ
  - ☑ Kernel

Model Selection: SVM for Regression 4

— SVM —



① $E_T = E_D + E_R$

→ Replaced by an ϵ-insensitive function:

② $$E_D = \begin{cases} 0, & \text{if } |y_{true} - y_p| < \epsilon \\ |y_{true} - y_p| - \epsilon, & \text{otherwise} \end{cases}$$

③ We minimize:

Kernel ε

$$C \sum_{i=1}^{N} \left[\left| y_{true_i} - y_{p_i} \right| - \epsilon \right] + 1/2 \, \frac{1}{N} \sum_{i=1}^{N} w_i^2$$

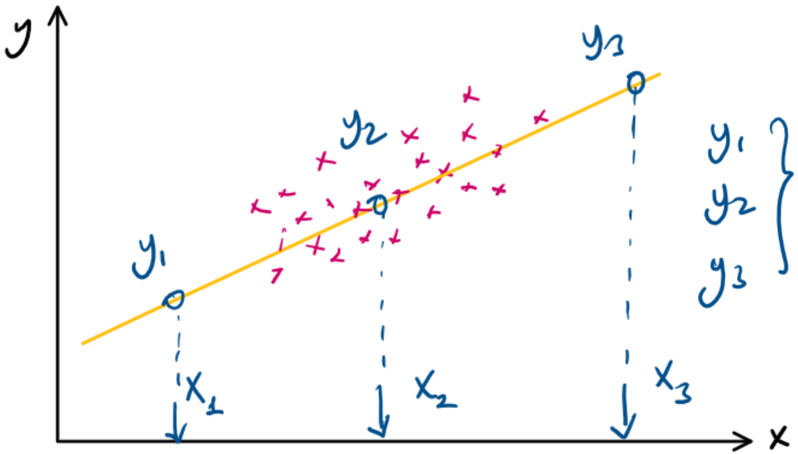
↓ Regularization parameter


Ateliers & Saveurs in Montreal

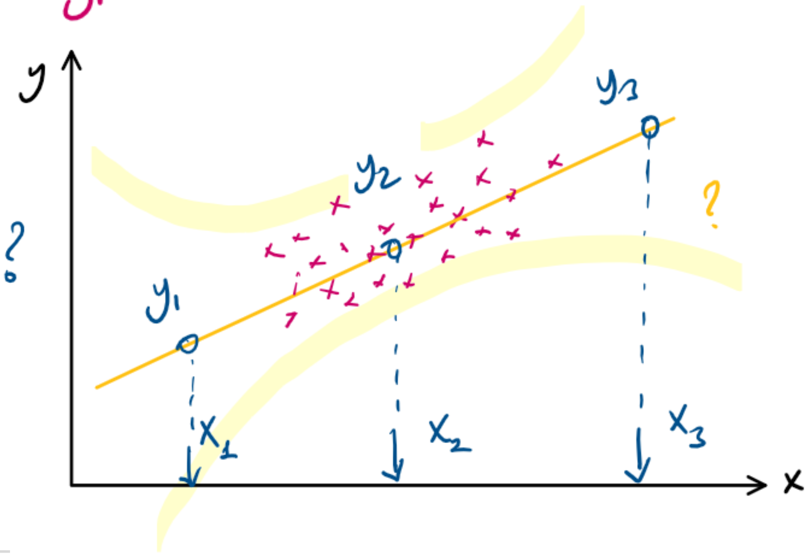
colab

Model Selection: Bayesian Regression 1

* Freq. based regression ⇒ fit w_i via error min. ⇒ y_p?
  ↳ predictions do not capture uncertainity → w_i → y_p



y_1, y_2, y_3 } how sure are you?



Model Selection: Bayesian Regression 2

① Bayesian approach; $\underbrace{y_t = y_p + \text{Error}}$

"Gaussian noise"

② $p(y_t | X, w, \alpha) = \mathcal{N}(y_t | y_p, \alpha) \Rightarrow \alpha$

"Given that"

③ $p(w | \lambda) = \mathcal{N}(w | 0, \lambda^{-1} I_p) \Rightarrow \lambda$

} Hyperparameters in scikit learn!

Bayes' Theorem

* "Hypothesis" + "evidence" = "New hypothesis"

Model Selection: Bayesian Regression 3

Bayes' Theorem

* "Hypothesis" + "evidence" = "New hypothesis"

Prior knowledge ⟹ Posterior knowledge

"probability"

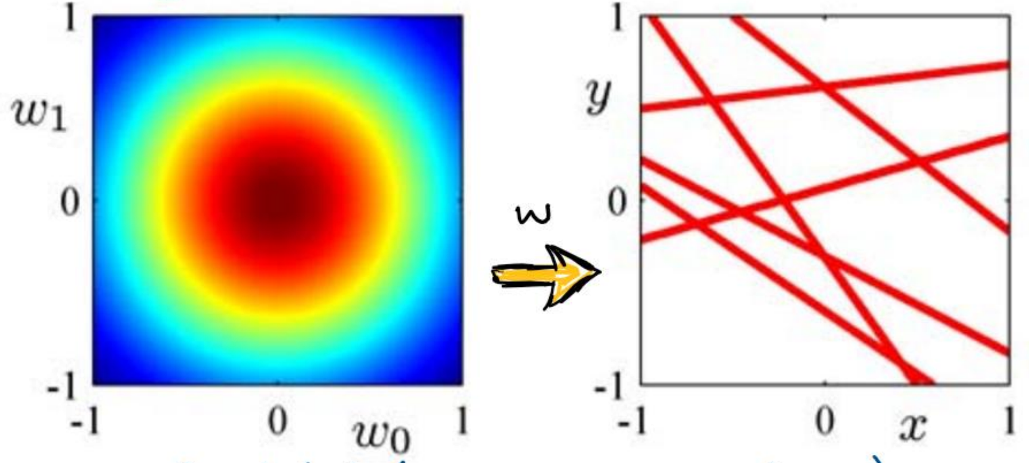
* $$\text{Posterior prob.} = \frac{\text{Likelihood} \times \text{Prior knowledge or prob.}}{\text{Evidence}}$$

Model Selection: Bayesian Regression 4

Before any data points observed:

Model:

$y = w_0 + w_1 x$



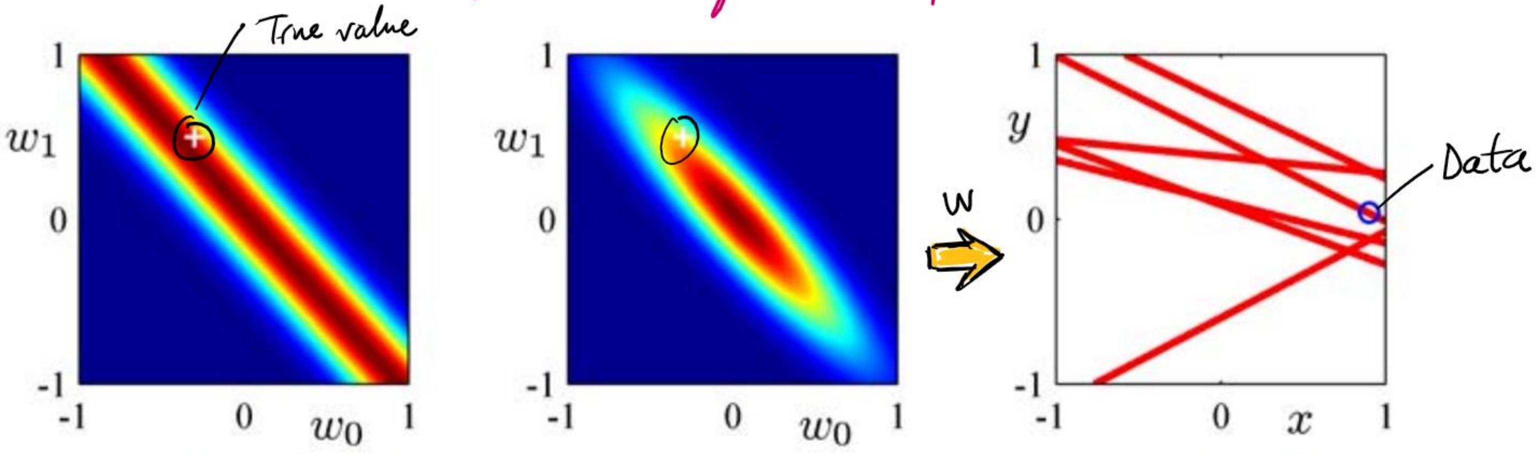
prior distribution of w

$y(x, w)$

Pattern Recognition and Machine Learning, Chapter 3

Model Selection: Bayesian Regression 4

After observing 1 data point:



Likelihood function: → Posterior (updated) probability

* $p(y_t | x, w)$
* $w \leftarrow$ "Prior"

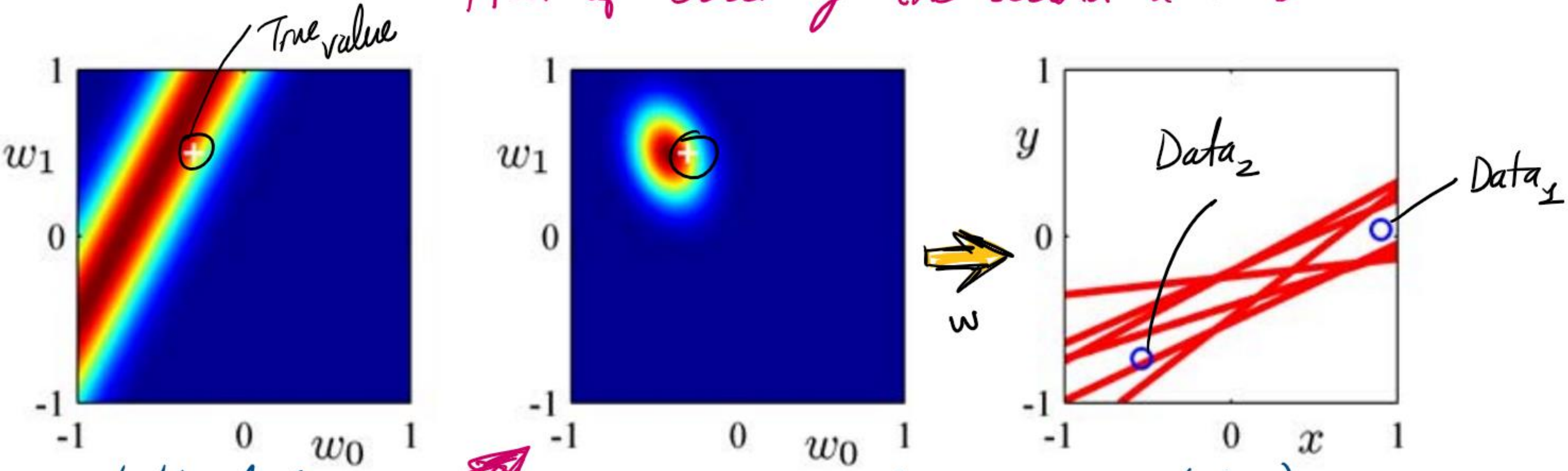
$y(x, w)$

* Lines are being accumulated around data.

Pattern Recognition and Machine Learning, Chapter 3

Model Selection: Bayesian Regression 4

Effect of Observing the second data:



Likelihood func.:
$p(y_t \mid x, w)$

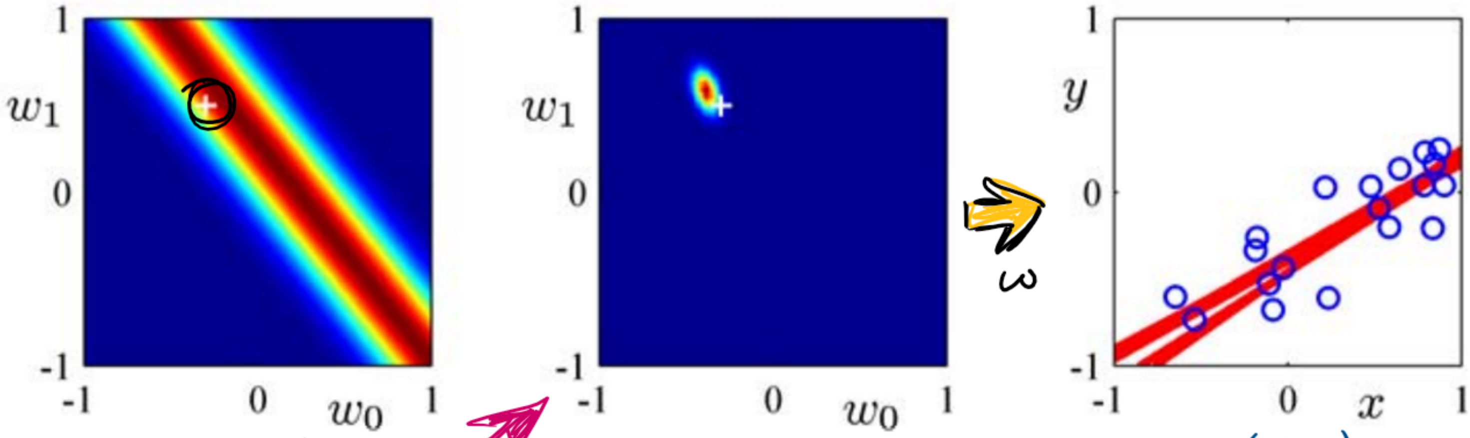
Posterior probability
(updated)

$y(x, w)$
* lines are accum.

Pattern Recognition and Machine Learning, Chapter 3

Model Selection: Bayesian Regression 4

Effect of observing 20 data points



Likelihood func.

Posterior probability (updated)

$y(x,w)$

* Lines are condensed.

Pattern Recognition and Machine Learning, Chapter 3

colab

Additional Notes