Input and Variable Selection for Local Models

Antti Sorjamaa

Outline

- Basic Concepts
- Input selection
- Models
  - *k*-NN, Lazy Learning
- Variable Selection
  - Leave-one-out, Bootstraps
- Results

Selection – The Word of Today

- Inputs
  - Input selection method (Wrapper or Filter)
- Model
- Parameters
  - Validation method
  - Bounds
- Local or Global
- Data sets
  - Learning, validation, test

Selection Principle

- Notations:

$$\mathbf{x}_n \in R^d, y_t \in R$$

$$\hat{y}_n = g(x_n, q)$$

$$\mathrm{E}_{training, validation}$$

- Generalization Error

$$\mathrm{E}_{gen}(q) = \lim_{N \to \infty} \sum_{n=1}^{N} \frac{\left(\mathrm{g}(\mathbf{x}_n, q) - y_n\right)^2}{N} \longrightarrow \boxed{\hat{\mathrm{E}}_{gen}(q)}$$

↑ to minimize

Input Selection

- Exhaustive method
  - "Brute force"
  - All 2^d input combinations explored
- Forward or Backward
  - Only $d(d-1)$ input sets evaluated
  - Local minima problems **à** Suboptimal
- Forward-Backward
  - "More Optimal" **ß** More input sets estimated
  - Time consumption unknown beforehand

Input Selection (2)

- Input selection with Forward-Backward method
  - Initialization

Possible Action	Selected Inputs 1	2	3	4	5
Initial	x	x			x
1		x			x
2	x				x
3	x	x	x		x
4	x	x		x	x
5	x	x			

k-NN

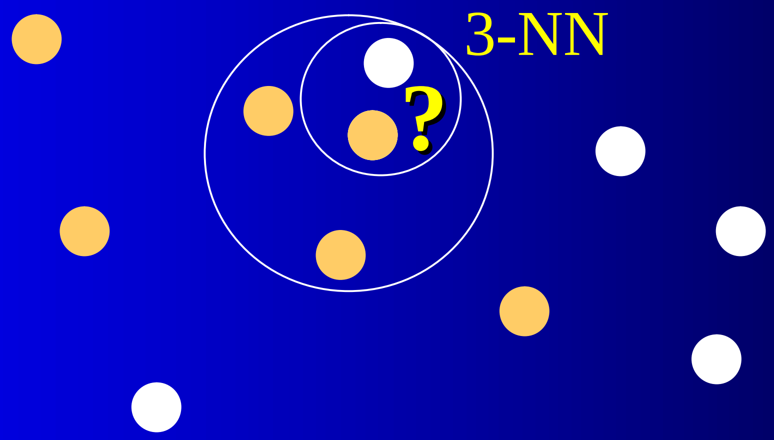
- Fast and reliable
- Can be used as a part or as a whole approximator
- Can be used with many different methods
- Only inputs and *k* need to be determined beforehand

k-NN

For Classification:

For Regression:

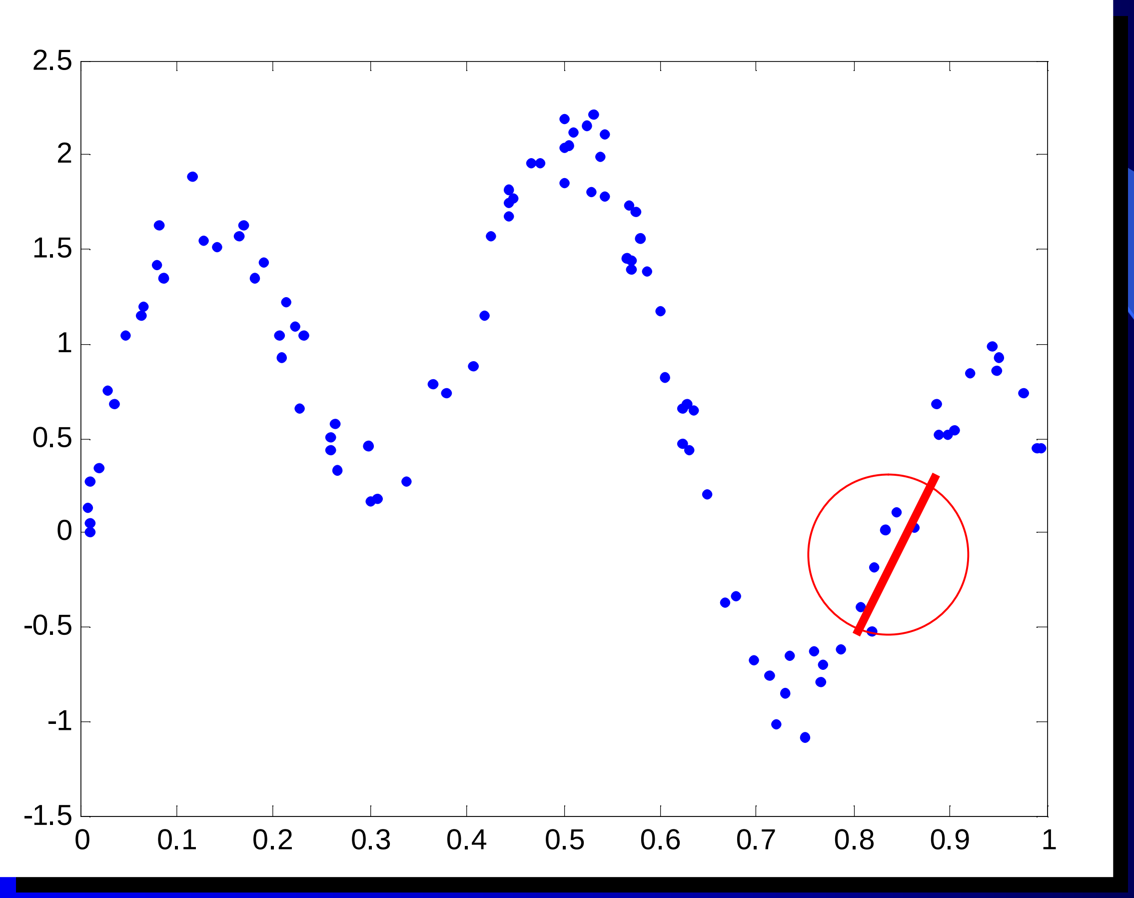
$$\hat{y}_i = \frac{\sum_{j=1}^{k} y_{\mathrm{P}(j)}}{k}$$



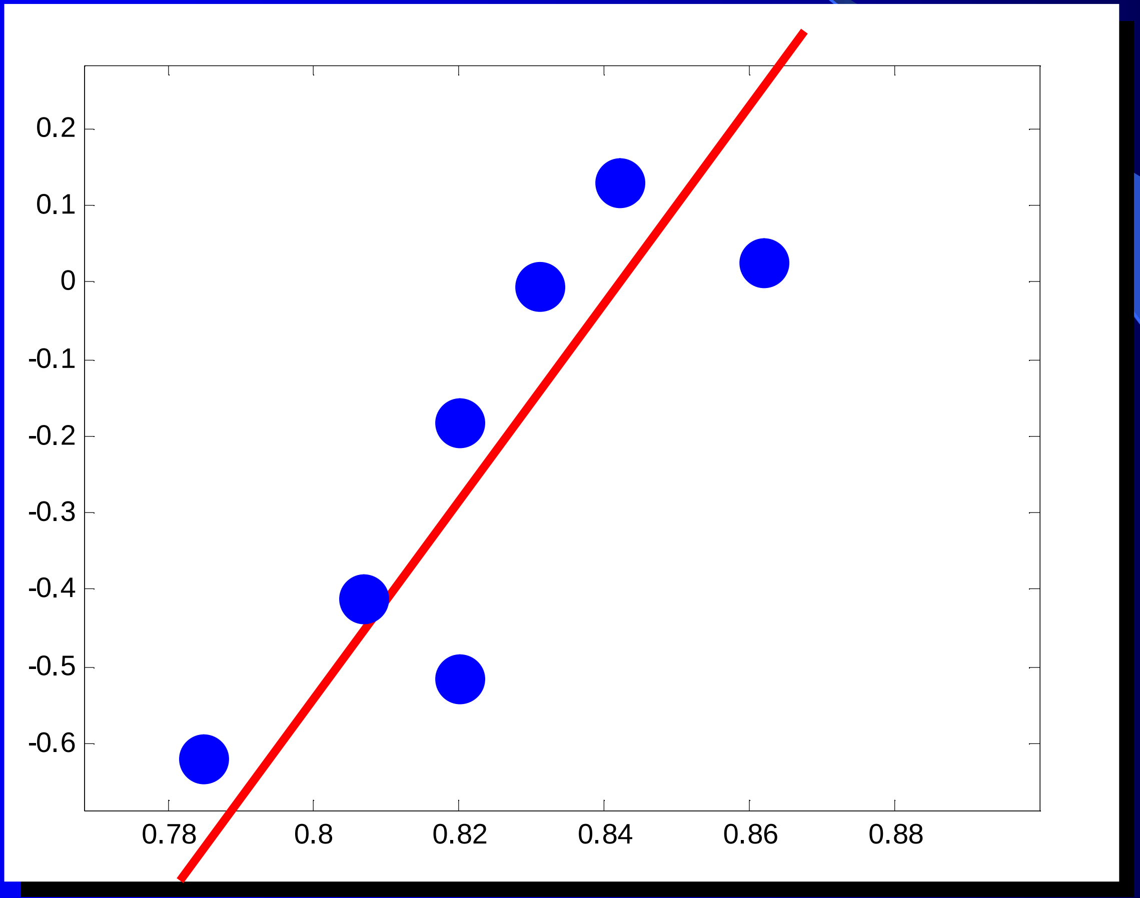
Lazy Learning

- Local, linear model
- Laziness
  - ”Do nothing until query”
  - No learning mandatory
- Compared to *k*-NN (local, constant model)
  - More time consuming than *k*-NN
  - Almost as diversified
- Locality can be ”globalized” incrementally

Local, linear model



Local, linear model



Formula

$$y_i = \mathrm{f}(\mathbf{x}_i) + e_i$$

$$\sum_{i=1}^{N} \{(y_i - \mathbf{x}_i'\boldsymbol{\beta})^2\, \mathrm{K}(\frac{\mathrm{d}(\mathbf{x}_i, \mathbf{x}_q)}{h})\}$$

Formula

$$\sum_{i=1}^{N}\{(y_i - \mathbf{x}_i'\boldsymbol{\beta})^2 \, \mathrm{K}(\frac{\mathrm{d}(\mathbf{x}_i,\mathbf{x}_q)}{h})\}$$

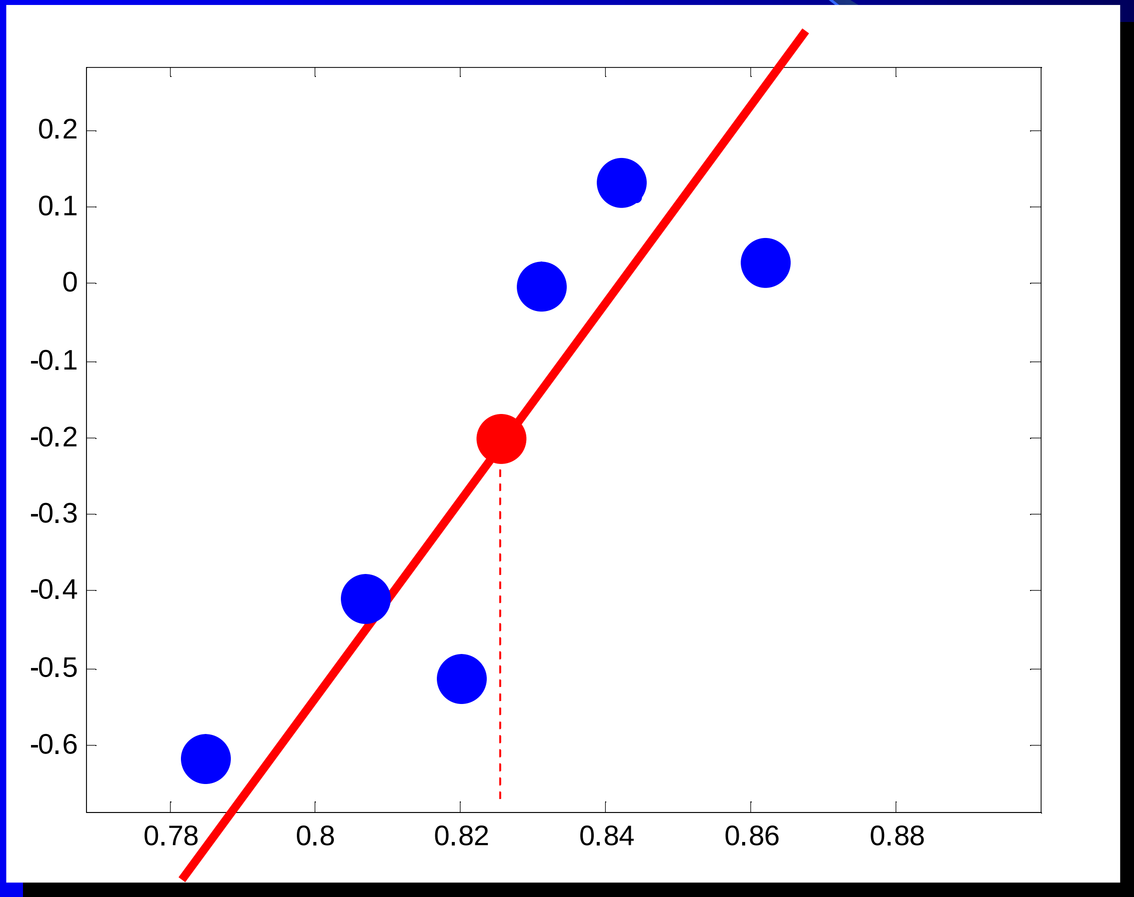
Simplified version: K **à** KNN

$$\mathbf{P} = (\mathbf{X}'\mathbf{X})^{-1}$$

$$\hat{\boldsymbol{\beta}} = \mathbf{P}\mathbf{X}'\mathbf{y}$$

$$\boxed{\hat{y}_q = \mathbf{x}_q'\hat{\boldsymbol{\beta}}}$$

“Do nothing until query”

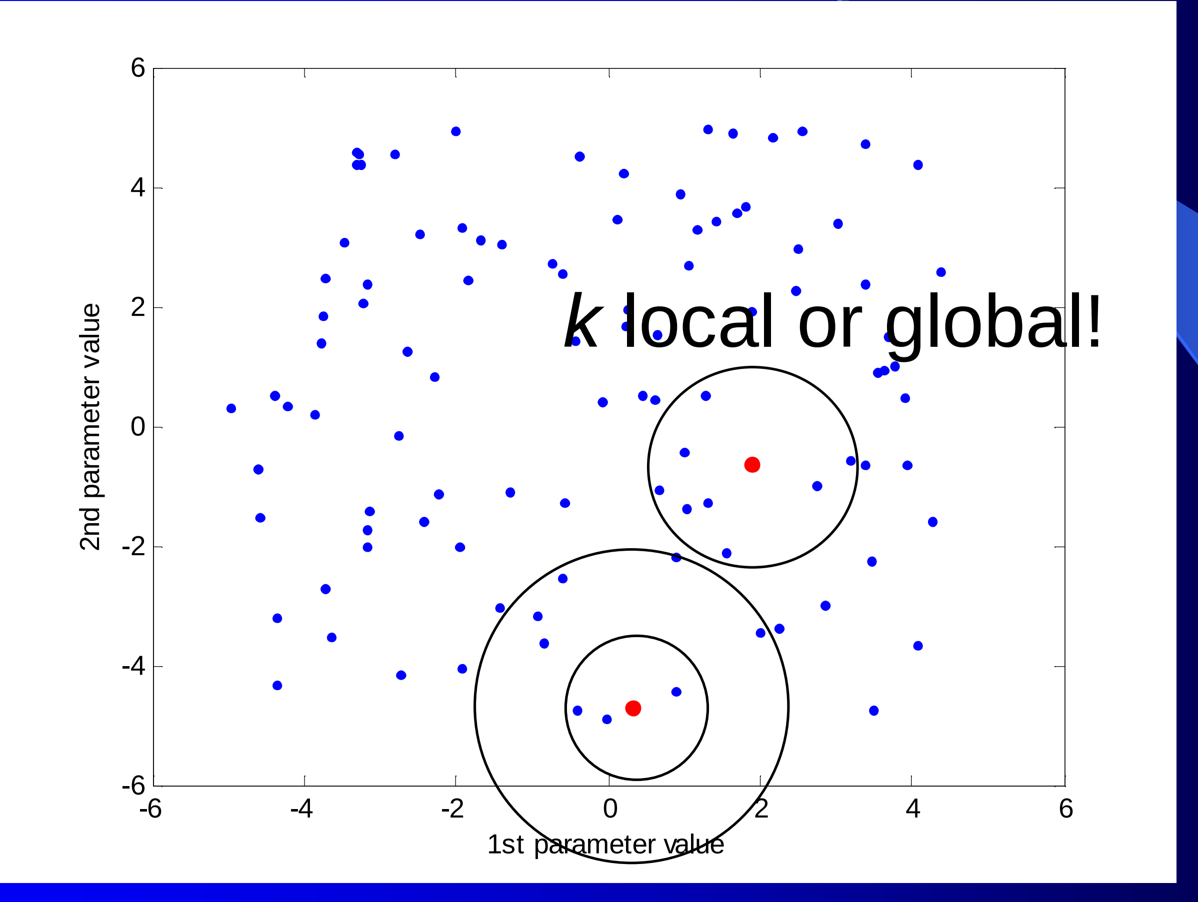


“Do nothing until query”

New input needs an output approximation

- Validate optimal inputs
  - Search nearest neighbors
  - Validate optimal neighborhood size
    - Build linear model
- Calculate the needed estimate

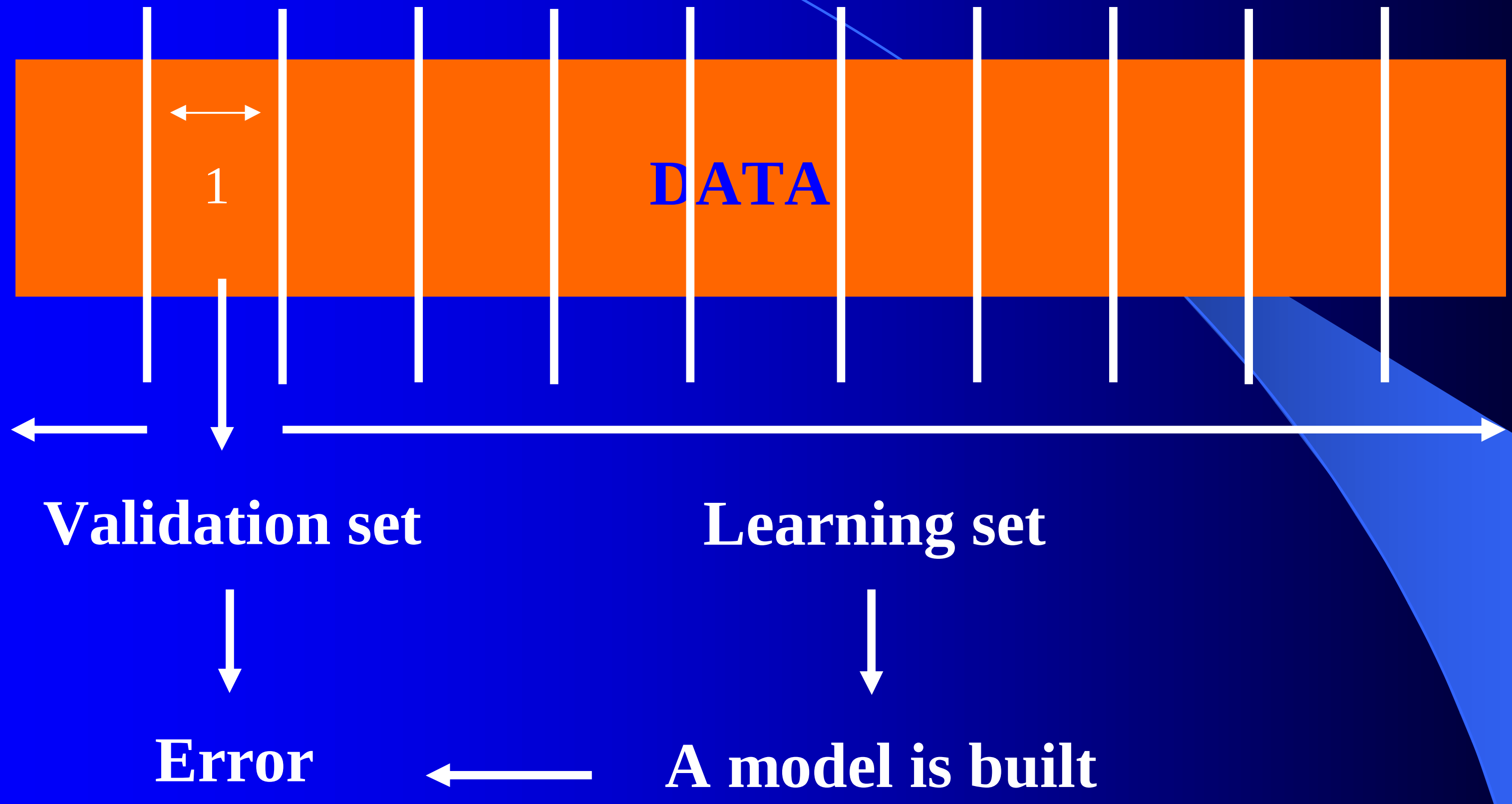
Example: $y(t)$=LL($y(t$-1),$y(t$-2))



Lazy Learning

- Locality can be ”globalized” incrementally
  - Global *k* instead of Local *k*
  - Globally selected inputs instead of Local
- More Globalization **à** Less Laziness
- Best amount of Globalization should be determined for each case
  - Intensive validation and testing
  - Different attached methods

Leave-One-Out (LOO)



Procedure repeated N times

$$\longrightarrow \hat{E}_{gen} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

Recursive formula for LOO

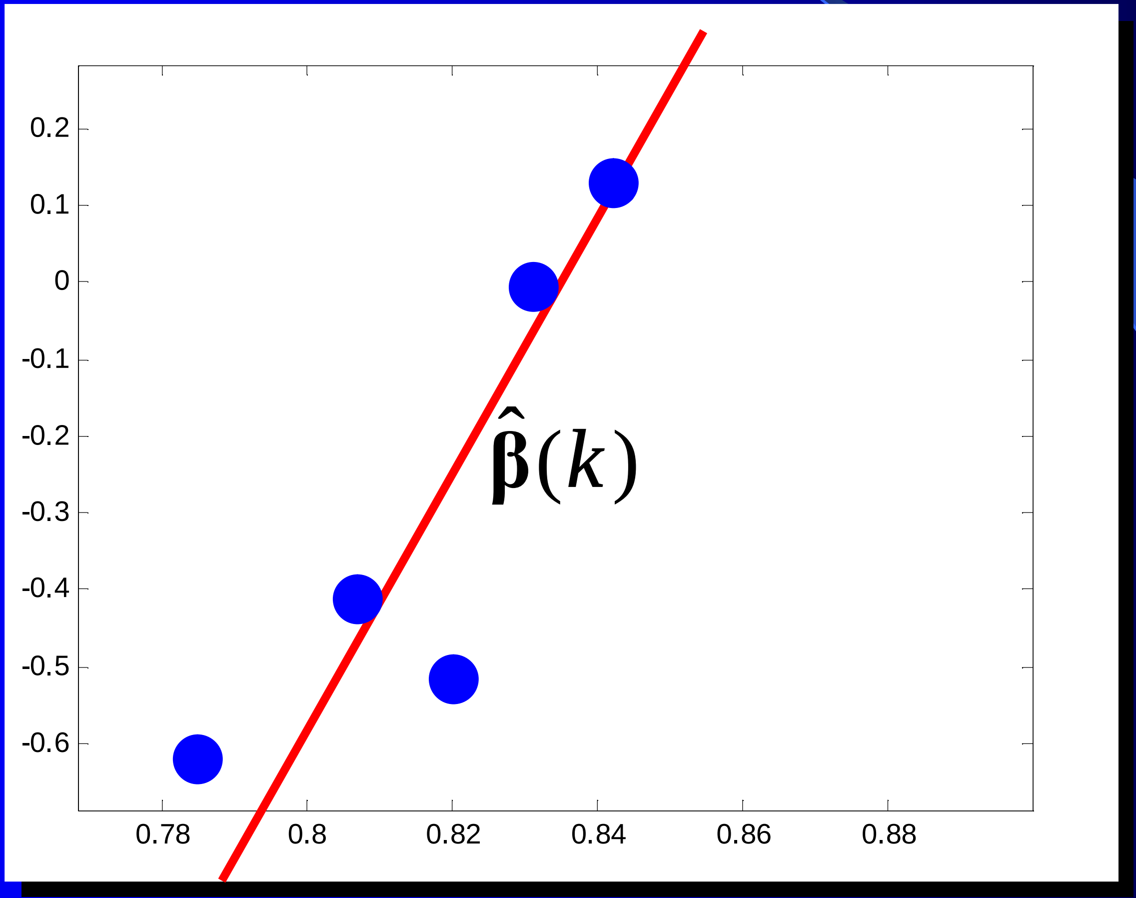
$$\mathbf{P}(k+1) = \mathbf{P}(k) - \frac{\mathbf{P}(k)\mathbf{x}(k+1)\mathbf{x}'(k+1)\mathbf{P}(k)}{1+\mathbf{x}'(k+1)\mathbf{P}(k)\mathbf{x}(k+1)}$$

$$g(k+1) = \mathbf{P}(k+1)\mathbf{x}(k+1)$$

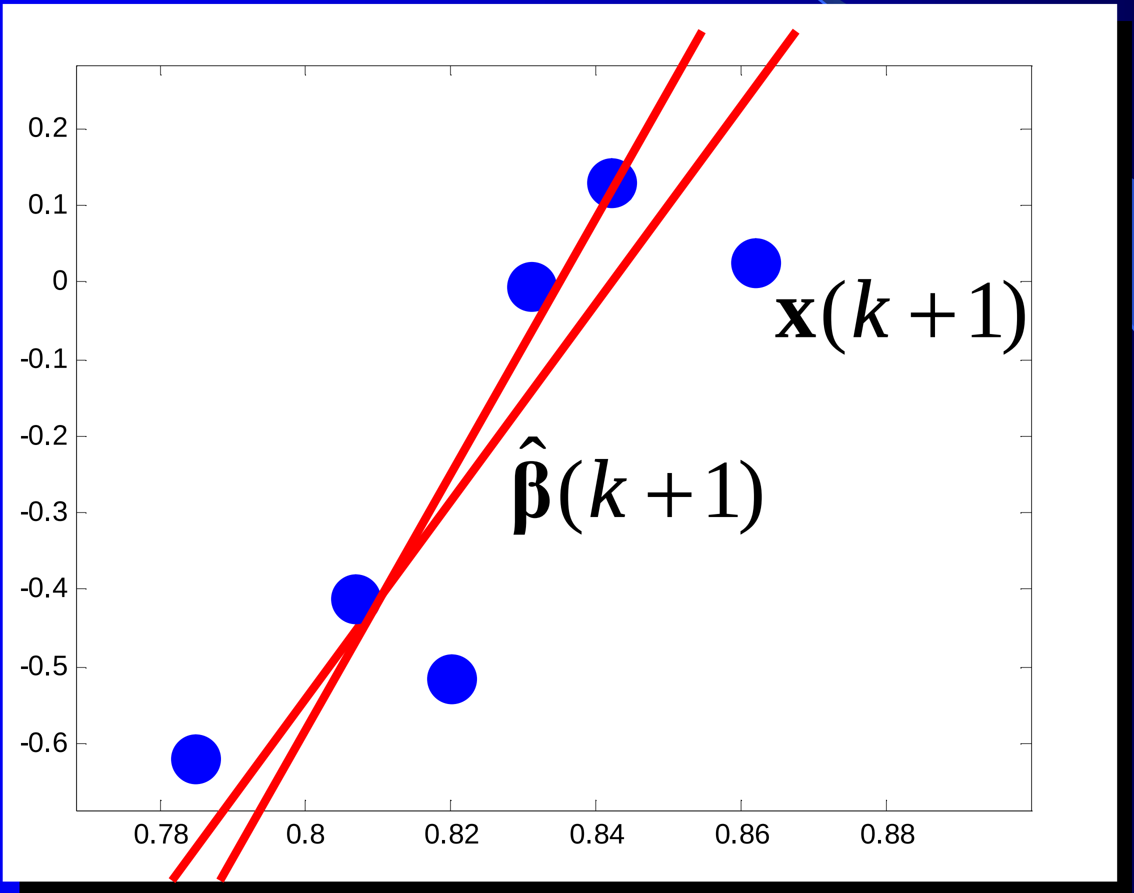
$$e(k+1) = y(k+1) - \mathbf{x}'(k+1)\hat{\boldsymbol{\beta}}(k)$$

$$\hat{\boldsymbol{\beta}}(k+1) = \hat{\boldsymbol{\beta}}(k) + g(k+1)e(k+1)$$

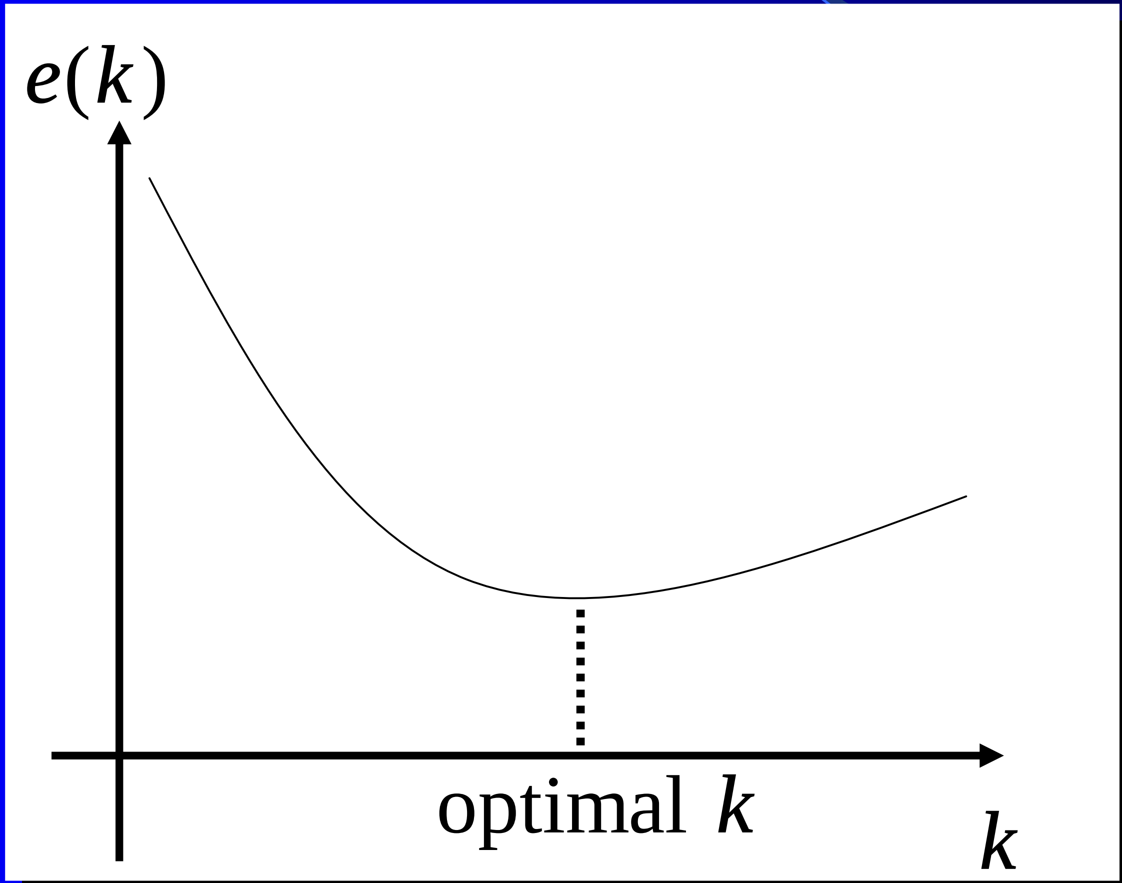
Recursive formula for LOO



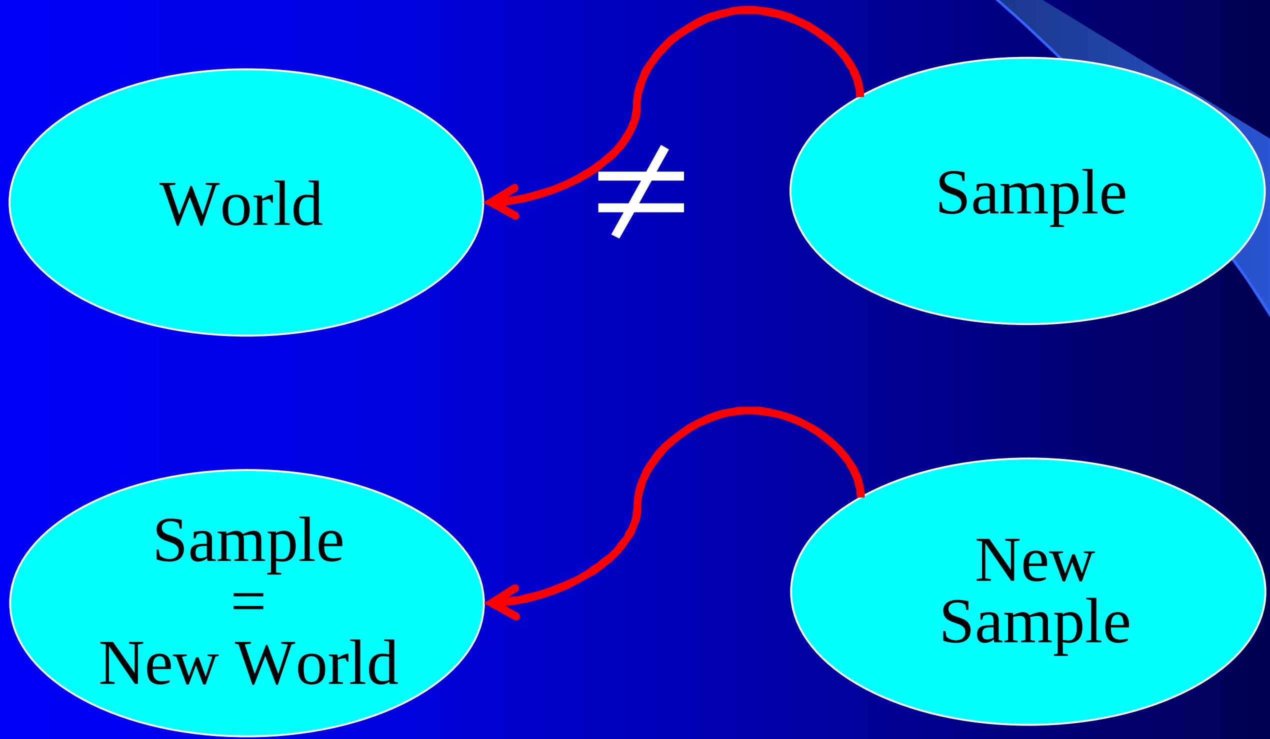
Recursive formula for LOO



Recursive formula for LOO



Bootstrap Resampling



Bootstrap Resampling

Definition: $\text{optimism}(q) \triangleq \mathrm{E}_{sam,gen}(q) - \mathrm{E}_{sam,sam}(q)$

$$\mathrm{E}_{sam,gen}(q) = \mathrm{E}_{sam,sam}(q) + \text{optimism}(q)$$

New World ← New sample

Estimate: $\text{opti}\hat{\text{m}}\text{ism}(q) = \frac{1}{B}\sum_{b=1}^{B}\left(\mathrm{E}^{b}_{new,sam}(q) - \mathrm{E}^{b}_{new,new}(q)\right)$

$$\hat{\mathrm{E}}_{sam,gen}(q) = \mathrm{E}_{sam,sam}(q) + \text{opti}\hat{\text{m}}\text{ism}(q)$$

Bootstrap 632

Bootstrap: $$\text{opti}\hat{\text{m}}\text{ism}(q) = \frac{1}{B}\sum_{b=1}^{B}\left(E^{b}_{new,sam}(q) - E^{b}_{new,new}(q)\right)$$

Bootstrap 632: $$\text{opti}\hat{\text{m}}\text{ism}^{632}(q) = \frac{1}{B}\sum_{b=1}^{B}\left(E^{b}_{\overline{new},new}(q)\right)$$

$$\overline{new} = sample - new$$

$$\hat{E}_{gen} = (1-0.632)\,E_{sam,sam} + 0.632\,\text{opti}\hat{\text{m}}\text{ism}^{632}(q)$$

- 0.632 is derived from probability of single data point to be selected to bootstrap set
- Unbiased and faster to evaluate

The Method

- Input selection with brute force
  - All 2^d input combinations explored
- Using *k*-NN as approximator
- *k* selected with Leave-one-out, Bootstrap and Bootstrap 632
  - Best *k* selected with each method
- Best input combination selected with each method

Results

- Darwin Sea Pressure Data – 1400 values
  - 1000 values for training and 400 for testing

	Selected Inputs	*k*	$\hat{E}_{gen}$	Test error
LOO	t - {1, 2, 3, 5, 7, 8}	15	0.9219	1.1650
Bootstrap	t - {1, 2, 4, 5, 7, 8}	1	0.6054	1.8458
Bootstrap 632	t - {1, 2, 3, 5, 7, 8}	16	0.9333	1.1625

The Method²

- Input selection
  - For k-NN, all 2^d input possibilities explored
  - For LL, Backward Selection and continuous
- *k* selected with Leave-one-out
  - Best *k* selected with each method combination
- Best input combination selected with each method combination
- Testing *k*-NN selected inputs with LL

Results²

- Santa Fe Data – 10 000 values
  - 1000 values for training and 9000 for testing

Method	Learning		Calculation time	Test	Prediction 40 steps
	k	LOO error	Minutes	MSE	MSE
LL	56	42.32	2.58	42.0746	1765.6
LL pruned	59	19.42	13.95	20.6037	148.37
k-NN	3	57.71	33.78	53.5387	1252.1
k-NN + LL	15	33.57	0.20	31.4548	1770.1

Conclusions

- Leave-one-out is fast and good method to select inputs
- Bootstraps can select more optimal number of neighbours for *k*-NN
- Inputs selected with *k*-NN are not as good to use with LL than the ones selected with LL
  - **à** *k*-NN is not good filter for LL

Questions?

Publications:

- A. Sorjamaa, A. Lendasse, and M. Verleysen, “Pruned Lazy Learning Models for Time Series Prediction,” pp. 509–514, ESANN 2005.
- A. Sorjamaa, N. Reyhani, and A. Lendasse, “Input and Structure Selection for *k*-NN Approximator,” in Lecture Notes in Computer Science, vol. 3512, pp. 985–991, IWANN 2005.
- Chris Atkeson, A. Moore and S. Schaal. Locally weighted learning, AI Review, 11:11-73, April 1997