Explainable machine learning in the low data regime

Guido von Rudorff, University of Kassel

vonrudorff@uni-kassel.de nablachem.org/talks

Supervised regression setting

We have

- Few data points (20 - 10.000)
- Noise / uncertainty estimate
- Scalar target

We want

- Non-parametric model: no fixed functional form
- Data driven: more data yields better model
- Explainable: understand answer

Animal	Typical Speed[1]
Horse	30 km/h
Parrot	35 km/h
Swallow	60 km/h
Python	3 km/h

1| All numbers in this talk are illustrative only.

Machine learning requires

- similarity (similar features = similar labels)
- coverage (data needs to be diverse and spanning query domain)

Animal	Legs	Wings	Weight	Speed
Horse	4	0	500 kg	30 km/h
Donkey	4	0	450 kg	25 km/h
Parrot	2	2	800 g	35 km/h
Macaw	2	2	1 kg	25 km/h
Sparrow	2	2	24 g	30 km/h
Swallow	2	2	20 g	60 km/h
Python	0	0	20 kg	3 km/h
Cobra	0	0	5 kg	12 km/h

Model form

Our case

- Features: Legs, Wings, Weight
- Target / Label: Speed

Features | Model parameters

$$f(\mathbf{x}|\mathbf{w}) = y \tag{1}$$

Model | Target

Training

- Chose model f for N data points
- Find parameters $\mathbf{w}$
- Estimate typical error

Features

- Maps real life to math: $\mathbf{x}$(Animal)
- Free to choose
- Similar target, similar representation
- Chemistry: 20-20.000 features

Kernel-Ridge-Regression (KRR)

Features of training points

$$\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

Kernel matrix element — Kernel function

Model weights — Training labels

$$\mathbf{w} = (\mathbf{K} + \lambda \mathbf{I}_N)^{-1} \mathbf{y} \tag{3}$$

Kernel matrix ($N \times N$) — Identity matrix ($N \times N$) — Regularization

Query — Training point features

$$\hat{y}(\mathbf{x}_q) = \sum_{i=1}^{N} w_i k(\mathbf{x}_i, \mathbf{x}_q) \tag{4}$$

Prediction — Weight of i-th training point

Kernel-Ridge-Regression (KRR) (cont'd)

⊞ Advantages

- Non-parametric, non-linear
- Cheap training, runs on CPU[1]
- Simple to implement (about 10 lines of code)
- Answer is closed form expression

⊟ Disadvantages

- Training memory $\mathcal{O}(N^2)$
- Training time $\mathcal{O}(N^3)$
- Inference memory/time $\mathcal{O}(N)$
- Needs a representation

1| 500 data points: full training including hyperparameter optimisation takes five seconds.

Radial Basis Functions (RBF kernels)

Choose kernel $k(\mathbf{x}_i, \mathbf{x}_j) = k(\|\mathbf{x}_i - \mathbf{x}_j\|) = k(d)$: 1d-function

Common kernels

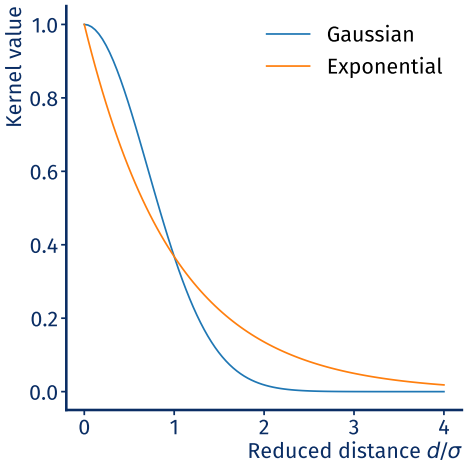
- Gaussian $\exp(-d^2/\sigma^2)$
- Exponential $\exp(-d/\sigma)$

Others

- see literature (about 15)
- need to satisfy conditions
- formal: match target smoothness

Hyperparameters

- Most: length scale σ
- Others: 2-4



Dealing with Uncertainty

We learn: $f(\mathbf{x})$

Uncertainty in label value (f)

- Measurement accuracy
- Variation in population

Individual	Duck speed
Donald	1 km/h
Dagobert	3 km/h
Track	2 km/h

Uncertainty in features (x)

- Incompleteness of observation
- Accuracy of features

Environment	Duck speed
Water	3 km/h
Land	3 km/h
Air	54 km/h

Kernel-Ridge-Regression

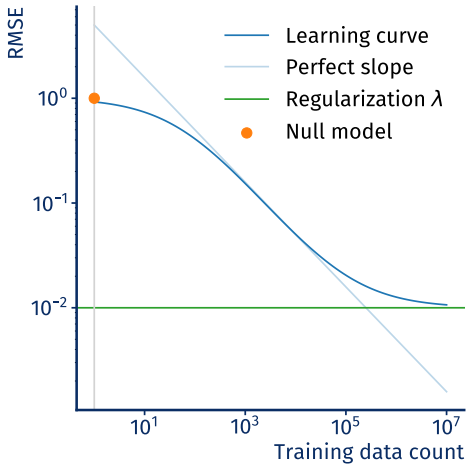
- noise increases regularization strength λ
- defines the limit of learning

How do I know my model works?

- Plot *Root mean squared error* (RMSE)
- Should decay as $N^{-1/2}$
- Limits
  - Above: *null model* (standard deviation)
  - Below: regularization λ

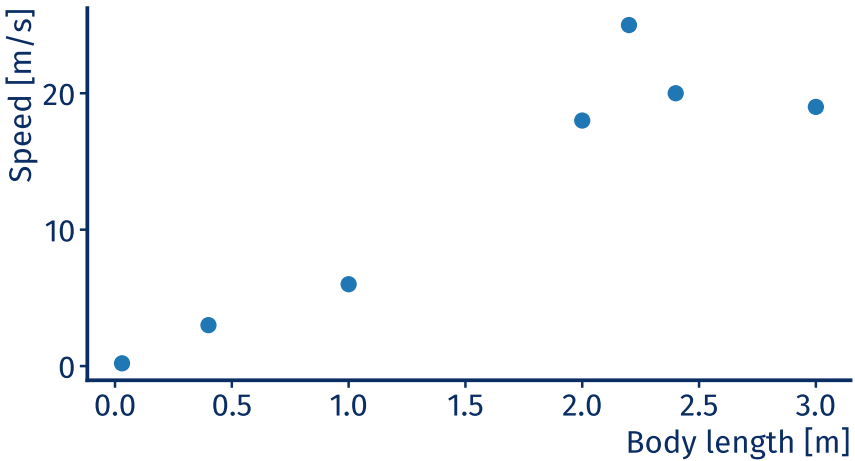
Diagnostic

- Flat (initial): irrelevant features or too many dimensions
- Flat (later): irrelevant features
- Upwards: incomplete hyperparameter optimisation
- Curving above λ: noise

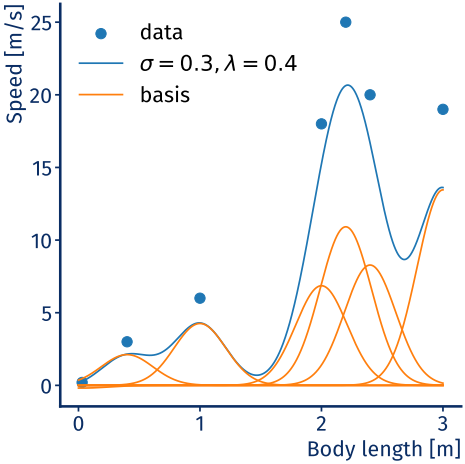


1D example

Fish speed as function of body length



One model



$$\underbrace{\hat{y}}_{\text{Prediction}}(\overbrace{\mathbf{x}_q}^{\text{Query}}) = \sum_{i=1}^{N} w_i k(\underbrace{\mathbf{x}_i}_{\text{Training point}}, \mathbf{x}_q) \quad (5)$$

Questions

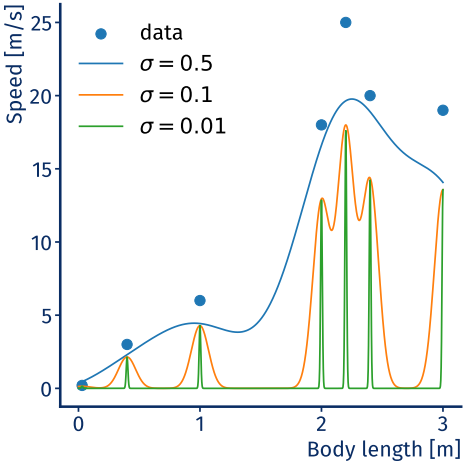
- good model?
- what should change?

Need to optimize hyperparameters

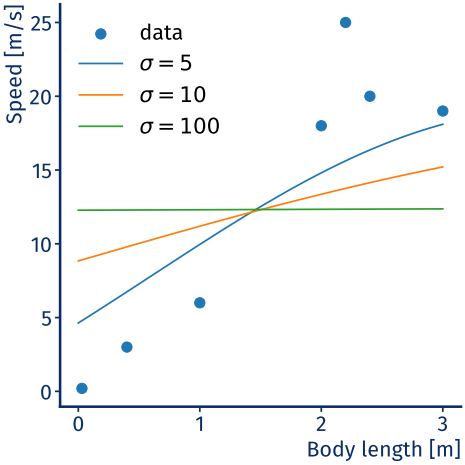
- length scale σ
- regularization λ

Length scale

Too small ($\sigma \ll d_{\text{NN}}$)
Predict 0 almost everywhere



Too big ($\sigma \gg d_{\text{NN}}$)
Predict label average



Length scale heuristic

$$\mathsf{K}_{ij} = \exp(-(d_{ij}/\sigma)^2) \tag{6}$$

Reduced distance

Kernel matrix element

Too big ($\sigma \gg d_{\mathsf{NN}}$)

- $\mathsf{K}_{ij} \approx 1$
- Predict label average

Too small ($\sigma \ll d_{\mathsf{NN}}$)

- $\mathsf{K} \approx \mathsf{I}$
- Predict 0 almost everywhere

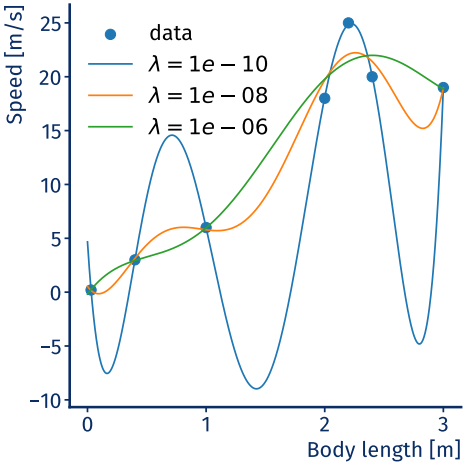
Characteristic length scale

- Median of nearest neighbors
- Refine on a grid
- Neighbors should interact / see each other

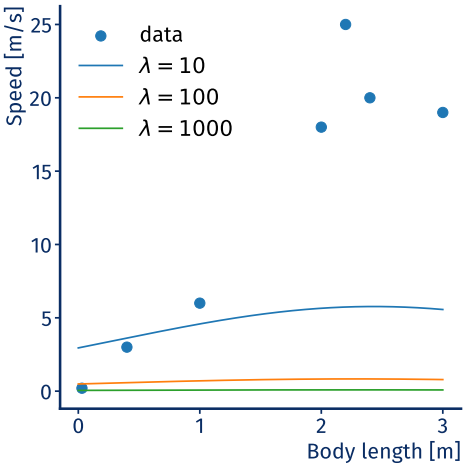
Centering

- Advisable to center labels to zero mean
- Only on training data (!)

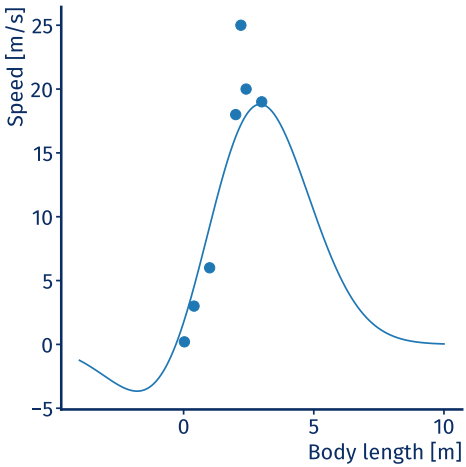
Too small ($\lambda \ll$ noise)
Overfit, oscillations



Too big ($\lambda \gg$ noise)
Underfit, predict 0



Extrapolation



Far away

- no support of kernel functions
- predicts 0: centering!

Close by

- oscillations from oversmoothing
- here: unphysical

Nested cross-validation

k-Fold crosvaliration

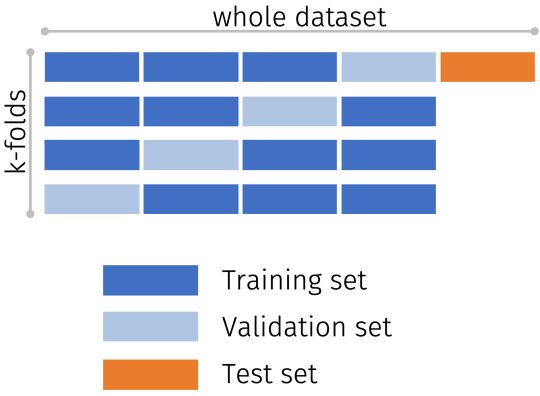
- Build k disjoint blocks
- Choose each block once as validation, train on rest
- Report metric on test set

⊞ Advantages

- Chooses all data points equally
- Converges quickly

⊟ Disadvantages

- Need to choose k
- Imbalanced blocks for most data sets
- Cannot reduce noise further



Errors as diagnostic

Training error $\mathcal{E}_{\text{tr}}$
RMSE of prediction on *training points*

Validation error $\mathcal{E}_{\text{val}}$
RMSE of prediction on *unused* training points

Test error / holdout error $\mathcal{E}_{\text{t}}$
RMSE of prediction on *unseen data*

Failure modes

- $\mathcal{E}_{\text{tr}} \gg \max(\lambda, \text{noise})$: underfitting, bad hyperparameters
- $\mathcal{E}_{\text{tr}} \ll \lambda$: overfit, regularization too weak
- $\mathcal{E}_{\text{t}} \gg \mathcal{E}_{\text{val}}$: train/test distribution mismatch or overfitting

Expected

- $\mathcal{E}_{\text{tr}} \approx \lambda$
- $\mathcal{E}_{\text{t}} \approx \mathcal{E}_{\text{val}}$

Baselines help

$$f_{\text{expensive}}(\mathbf{x}) = f_{\text{cheap}}(\mathbf{x}) + g(\mathbf{x}) \quad (7)$$

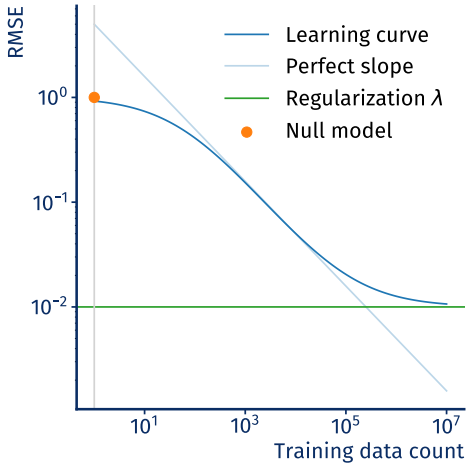
Alternative learning target

Shifts learning curve

- Model at $N = 1$: label standard deviation
- Nonlinear model can focus on hard part

Detrending

- Easy way to center labels
- Superimposes clusters in data
- Indication: structured / multimodal / skewed residuals



```
from sklearn.kernel_ridge import KernelRidge
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import root_mean_squared_error

X, y = ...
X_train, X_test, y_train, y_test = train_test_split(X, y)

grid = GridSearchCV(KernelRidge(kernel="rbf"),
    {"alpha": [1e-3, 1e-2, 1e-1, 1.0],  # role of lambda on the slides
     "gamma": [0.01, 0.1, 1.0, 10.0]},  # role of sigma on the slides
    cv=5
)
grid.fit(X_train, y_train)

model = grid.best_estimator_
rmse = root_mean_squared_error(y_test, model.predict(X_test))
```

Sample code, needs work for research.

Summary

Kernel-Ridge-Regression (KRR)
Non-parametric regression method

Radial Basis Functions (RBF)
Kernel functions that depend only on distance between feature vectors: effective length σ

Length scale σ
near median nearest-neighbor distance

Regularization λ
Prevents overfitting

Learning curves
RMSE vs training data size should decay as $N^{-1/2}$

Cross-validation
required to optimize hyperparameters and estimate generalization error without overfitting to training set

Extrapolation limitations
RBF kernels often predict centered mean far from training data

Baseline models
Cheap approximations can and should be subtracted from targets

Diagnostic errors
Training, validation, and test errors help understanding models