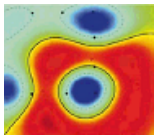
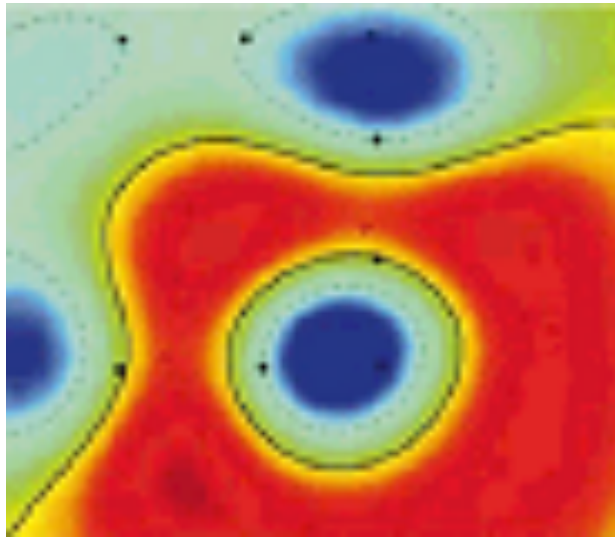


WiSe 2024/25

Machine Learning 1/1-X

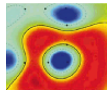
Technische Universität Berlin



Lecture 13 | Regression

Outline

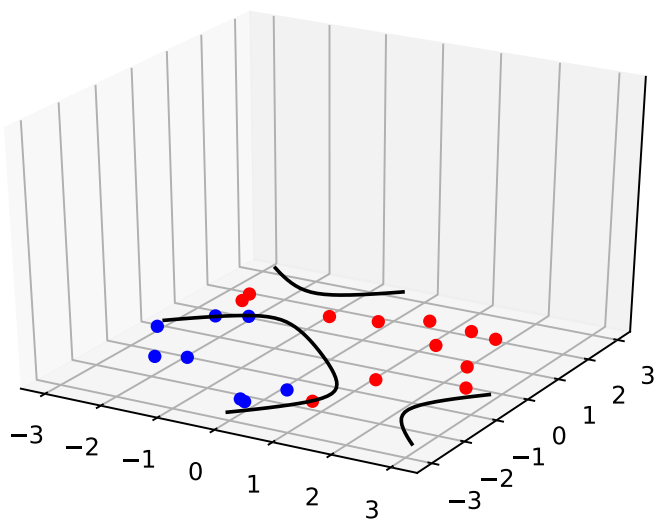
- Classification vs. Regression
- Regression:
  - Square Loss
  - Least Square Regression
  - Ridge Regression
  - Kernel Ridge Regression
  - Gaussian Processes
  - Robust Regression



Classification vs. Regression

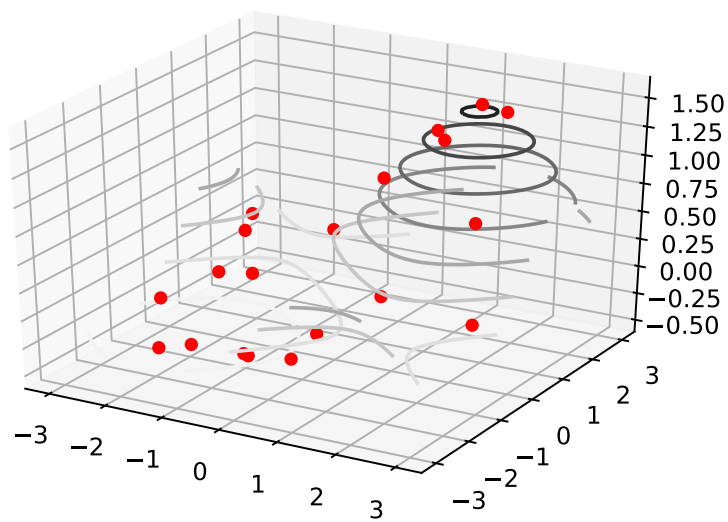
Classification

$f : \mathbb{R}^d \to \{-1, 1\}$



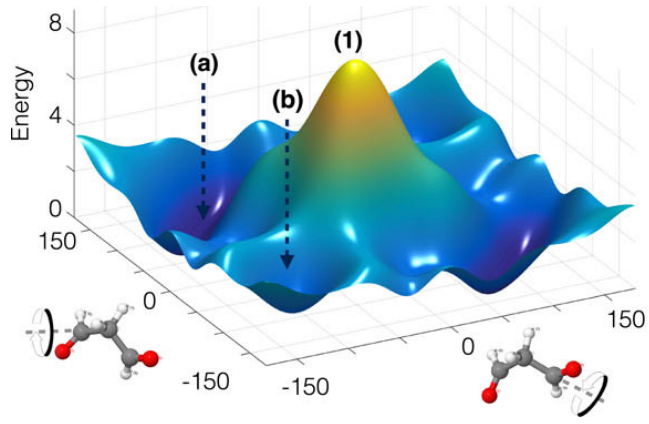
Regression

$f : \mathbb{R}^d \to \mathbb{R}$



Why Regression?

Example: Modeling physical systems

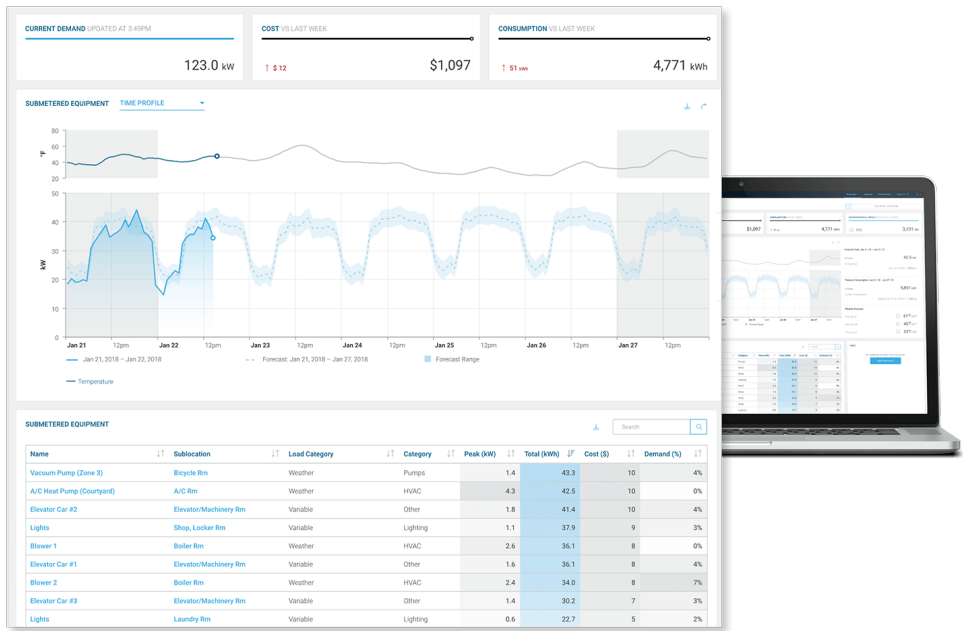


Source: Sauceda et al. J. Chem. Phys. 150, 114102 (2019)

- Useful models for molecular dynamics / chemical reactions.
- Quantity to predict (energy) is intrinsically real-valued.
- From the energy, one can derive quantities such as forces on different atoms.

Why Regression?

Example: Energy Foreasting



Source: https://www.enertiv.com/resources/faq/what-is-energy-forecasting

- Schedule energy-consuming tasks intelligently to minimize costs.
- Related tasks: demand forecasting, etc.

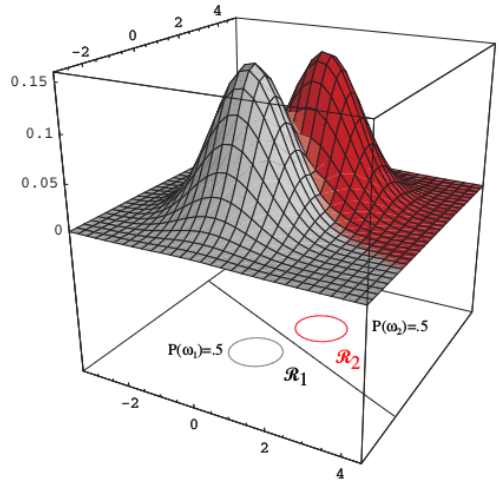
Recap: Optimal Classifier

Example of data generating assumption:

- Assume our data is generated for each class ω_j according to the multivariate Gaussian distribution

$$p(\boldsymbol{x}|\omega_j) = \mathcal{N}(\boldsymbol{\mu}_j, \Sigma)$$

and with class priors $P(\omega_j)$.



Resulting optimal classifier:

- The Bayes optimal classifier is derived as

$$\begin{aligned}
&\arg\max_j \{P(\omega_j|\boldsymbol{x})\} \\
&\quad = \arg\max_j \left\{\boldsymbol{x}^\top \Sigma^{-1} \boldsymbol{\mu}_j^\top - \frac{1}{2}\boldsymbol{\mu}_j \Sigma^{-1} \boldsymbol{\mu}_j + \log P(\omega_j)\right\}
\end{aligned}$$

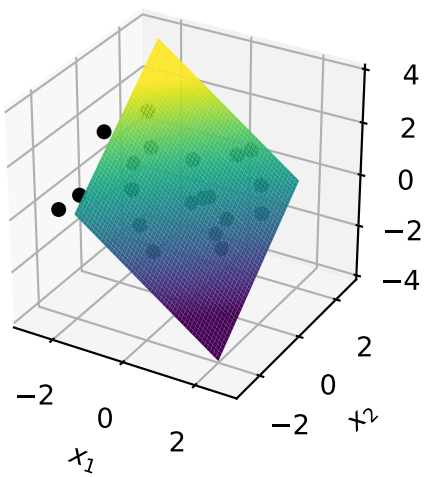
which is a linear classifier.

Building an Optimal Regressor

Example of data generating assumption:

- Let $\boldsymbol{x}$ and t denote the input and the output (target), respectively. Assume our data comes from the joint distribution

$$P(\boldsymbol{x}, t) = \mathcal{N}\Big(\underbrace{\begin{pmatrix} \boldsymbol{\mu}_{\boldsymbol{x}} \\ \mu_t \end{pmatrix}}_{\boldsymbol{\mu}}, \underbrace{\begin{pmatrix} \Sigma_{\boldsymbol{xx}} & \Sigma_{\boldsymbol{x}t} \\ \Sigma_{t\boldsymbol{x}} & \Sigma_{tt} \end{pmatrix}}_{\Sigma} \Big)$$



Resulting optimal regressor:

- Using identities of multivariate Gaussians, we get the conditional expectation:

$$\mathbb{E}[t|\boldsymbol{x}] = \mu_t + \underbrace{\Sigma_{t\boldsymbol{x}}\Sigma_{\boldsymbol{xx}}^{-1}}_{\boldsymbol{w}}\boldsymbol{x} + \underbrace{-\Sigma_{t\boldsymbol{x}}\Sigma_{\boldsymbol{xx}}^{-1}\boldsymbol{\mu}_x}_{b}$$

  This is best possible regressor in the sense of expectation, and again a linear model.

Objective-based learning

- In real-world applications, data generating distributions are not known, or are difficult to estimate.
- For practical purposes, we consider instead a class of models of limited complexity (e.g. linear)

$$y = \boldsymbol{w}^\top \boldsymbol{x} + b$$

with $\theta = (\boldsymbol{w}, b)$, and find the model in that class that minimizes the error on the given dataset $\mathcal{D}$ (empirical error):

$$\theta^\star = \arg\min_{\theta \in \Theta} \; R_{\text{emp}}(\theta, \mathcal{D})$$

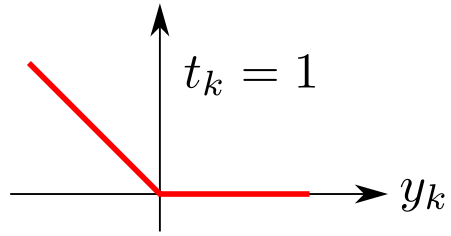
- Example of such approach in the context of classification: the *perceptron*.

Recap: The Perceptron

- Proposed by F. Rosenblatt in 1958.
- Classifier that prefectly separates training data (if the data is linearly separable).
- Trained using a simple and cheap iterative procedure.

The perceptron can also be seen as a gradient descent of the error function

$$\mathcal{E}(\boldsymbol{w}, b) = \frac{1}{N} \sum_{k=1}^{N} \max(0, -y_k t_k)$$



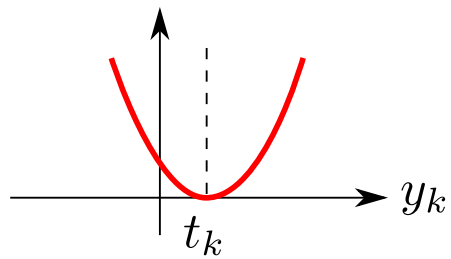
where $t_k \in \{-1, 1\}$ and $\max(0, -y_k t_k)$ is called the *hinge loss*.

Hinge Loss vs. Square Loss

Idea:

- Use a loss function designed for regression, e.g. the square loss:

$$\mathcal{E}(\boldsymbol{w}, b) = \frac{1}{N} \sum_{k=1}^{N} (y_k - t_k)^2$$



- The square loss $(y_k - t_k)^2$ is a popular way of measuring the error of a model and optimize it. [Gauss 1809; Legendre 1805]

C. F. Gauss
(1777–1855)

A. M. Legendre
(1752–1833)

Least Square Regression

Initial Trick

To simplify the derivations, Replace the bias by adding a constant dimension into the data and a corresponding weight:

$$\boldsymbol{x}^\top \boldsymbol{w} + b = \underbrace{[\boldsymbol{x}, 1]}_{\boldsymbol{x}}^\top \underbrace{[\boldsymbol{w}, b]}_{\boldsymbol{w}}$$

Therefore, the error function

$$\mathcal{E}(\boldsymbol{w}, b) = \frac{1}{N} \sum_{k=1}^{N} (\boldsymbol{w}^\top \boldsymbol{x}_k + b - t_k)^2$$

can be rewritten more compactly as:

$$\mathcal{E}(\boldsymbol{w}) = \frac{1}{N} \sum_{k=1}^{N} (\boldsymbol{w}^\top \boldsymbol{x}_k - t_k)^2$$

Least Square Regression

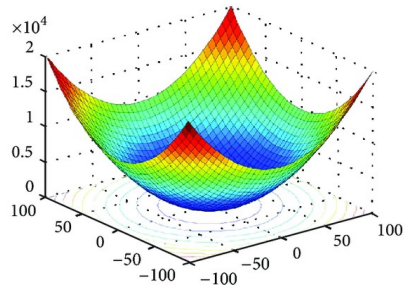
The error function to minimize can be developed as:

$$\begin{aligned}\mathcal{E}(\boldsymbol{w}) &= \frac{1}{N}\sum_{k=1}^{N}(\boldsymbol{w}^\top \boldsymbol{x}_k - t_k)^2 \\ &= \frac{1}{N}\sum_{k=1}^{N}\boldsymbol{w}^\top \boldsymbol{x}_k \boldsymbol{x}_k^\top \boldsymbol{w} - 2\boldsymbol{w}^\top \boldsymbol{x}_k t_k + \text{cst.} \\ &= \frac{1}{N}\boldsymbol{w}^\top XX^\top \boldsymbol{w} - \frac{2}{N}\boldsymbol{w}^\top X\boldsymbol{t} + \text{cst.}\end{aligned}$$

where $X = (\boldsymbol{x}_1|\ldots|\boldsymbol{x}_N)$ is a matrix of size $d \times N$ containing the data and $\boldsymbol{t} = (t_1, \ldots, t_N)$ is a vector of size N containing the targets.

Least Square Regression

Because $XX^\top$ is positive semi-definite, and the error $\mathcal{E}(\boldsymbol{w})$ is consequently a convex function of $\boldsymbol{w}$, a necessary condition for a minimum of the function is to satisfy $\nabla\mathcal{E}(\boldsymbol{w}) = \mathbf{0}$.

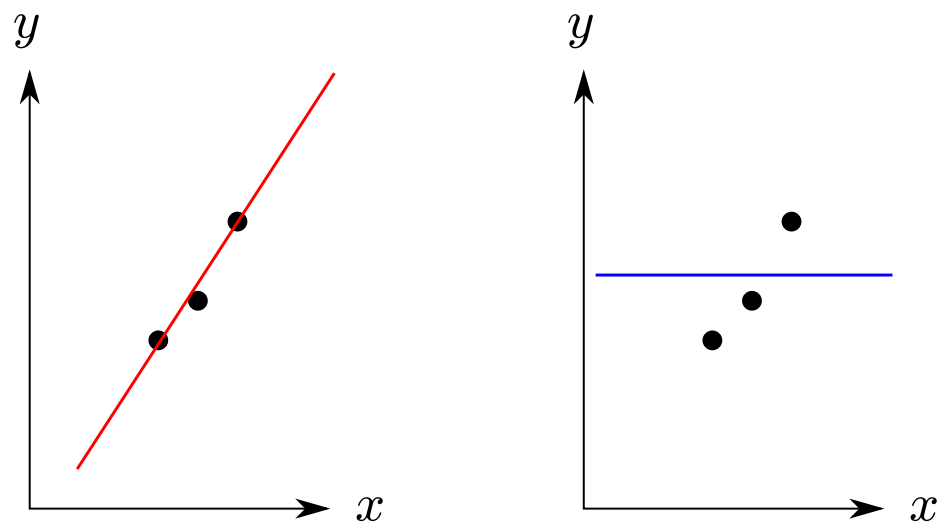


$$\begin{aligned}\nabla\mathcal{E}(\boldsymbol{w}) &= \nabla\left(\frac{1}{N}\boldsymbol{w}^\top XX^\top\boldsymbol{w} - \frac{2}{N}\boldsymbol{w}^\top X\boldsymbol{t} + \text{cst.}\right)\\ &= \frac{2}{N}XX^\top\boldsymbol{w} - \frac{2}{N}X\boldsymbol{t} = 0\end{aligned}$$

Rearranging terms and multiplying by $(XX^\top)^{-1}$ on both sides, we get the closed form solution:

$$\boxed{\boldsymbol{w} = (XX^\top)^{-1}X\boldsymbol{t}}$$

Least Square Regression, Dilemma



Questions:

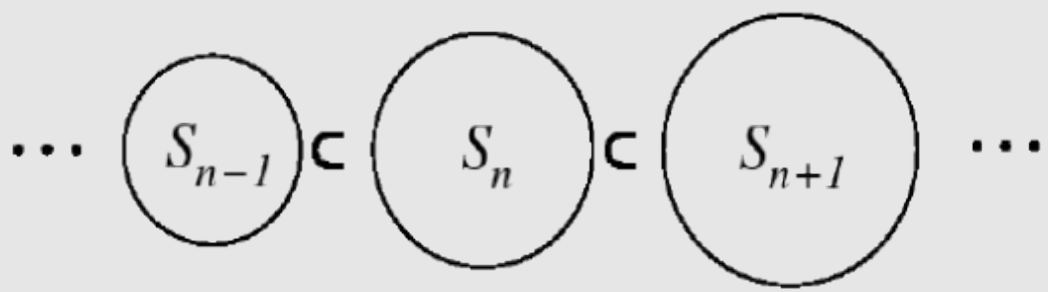
- Are variation in training data signal or noise?
- How well will the model generalize to new data?
- Can we learn models subject to the constraint that it has to be simple (e.g. of limited slope)?

Recap: Structural Risk Minimization (SRM)

Structural risk minimization (Vapnik and Chervonenkis, 1974) is an approach to measure complexity and perform model selection.

SRM Idea:

- Structure the space of solutions into a nesting of increasingly large regions.



- If two solutions fit the data, prefer the solution that also belongs to the smaller regions.

Ridge Regression

Idea:

- Implement the SRM principle by further restricting the class of functions the model can be selected from, i.e. $\min_{\boldsymbol{w}} \mathcal{E}(\boldsymbol{w}) \quad \text{s.t.} \quad \|\boldsymbol{w}\|^2 \leq C$.
- This objective can be minimized using Lagrange multipliers:

$$\begin{aligned}&\nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, \lambda) \\ &\quad = \nabla_{\boldsymbol{w}} \left(\frac{1}{N} \boldsymbol{w}^\top X X^\top \boldsymbol{w} - \frac{2}{N} \boldsymbol{w} X \boldsymbol{t} + \lambda \cdot (\|\boldsymbol{w}\|^2 - C) \right) = \mathbf{0}\end{aligned}$$

  which leads to:

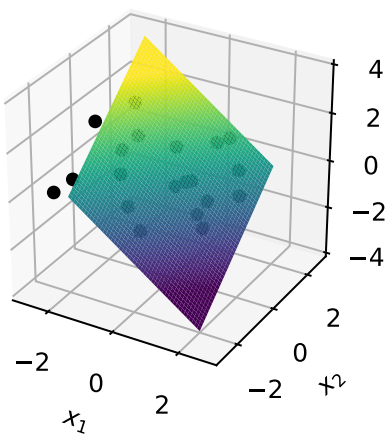
$$\boxed{\boldsymbol{w} = (X X^\top + \underbrace{N\lambda}_{\lambda} I)^{-1} X \boldsymbol{t}}$$

  where λ is chosen to minimize the error subject to the constraint being satisfied. This regularized least square is called *ridge regression*.
- In practice, instead of specifying the parameter C, it is common to directly treat λ as the hyperparamter.

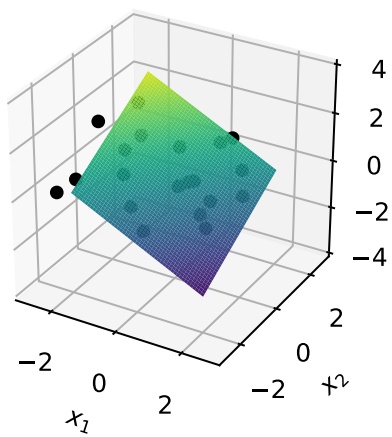
Ridge Regression

Example:

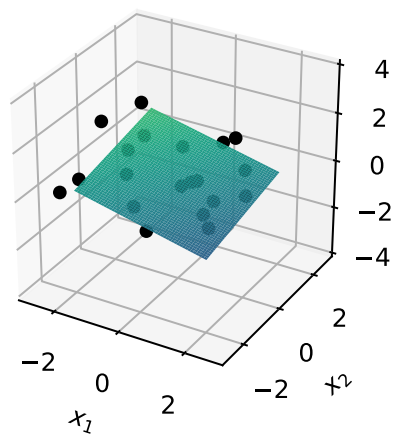
$\lambda = 0$



$\lambda = 5$



$\lambda = 25$

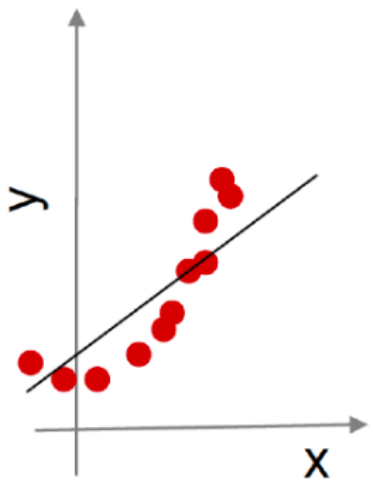


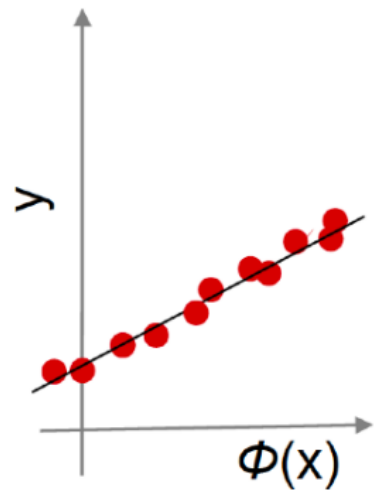
- The higher the parameter λ, the flatter the predicted function.
- High values of λ are desirable for noisy high-dimensional data.

From Linear to NonLinear Regression

Idea:

- If the function to predict is nonlinear, nonlinearly transform the input data $\boldsymbol{x}$ via some feature map $\Phi : \mathbb{R}^d \to \mathbb{R}^h$, and solve the problem linearly in the feature space.
- The feature map needs to be chosen appropriately.





Kernel Ridge Regression

- **Idea:** Redefine the prediction function $y = \boldsymbol{w}^\top \Phi(\boldsymbol{x})$ where $\boldsymbol{w} \in \mathbb{R}^h$ and the objective to minimize as

$$\mathcal{E}(\boldsymbol{w}) = \frac{1}{N} \sum_{k=1}^{N} (\boldsymbol{w}^\top \Phi(\boldsymbol{x}_k) - t_k)^2 \qquad \text{subject to} \quad \|\boldsymbol{w}\|^2 \leq C.$$

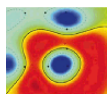
- The solution to this optimization problem is given by

$$\boldsymbol{w} = (\Phi(X)\Phi(X)^\top + \lambda I)^{-1} \Phi(X)\boldsymbol{t}$$

  where $\Phi(X) = (\Phi(\boldsymbol{x}_1)| \ldots |\Phi(\boldsymbol{x}_N))$, and for an appropriate choice of parameter λ.

- A new data point is then predicted as

$$\begin{aligned} y = \boldsymbol{w}^\top \Phi(\boldsymbol{x}) &= \Phi(\boldsymbol{x})^\top \boldsymbol{w} \\ &= \Phi(\boldsymbol{x})^\top (\Phi(X)\Phi(X)^\top + \lambda I)^{-1} \Phi(X)\boldsymbol{t} \end{aligned}$$



Kernel Ridge Regression

from: $y = \Phi(\boldsymbol{x})^\top (\Phi(X)\Phi(X)^\top + \lambda I)^{-1} \Phi(X)\boldsymbol{t}$

we arrive at: $\boxed{y = k(\boldsymbol{x}, X)(K + \lambda I)^{-1}\boldsymbol{t}}$

Kernel Ridge Regression

Observation:

- Predictions of the kernel ridge regression model:

$$y(\boldsymbol{x}) = k(\boldsymbol{x}, X)(K + \lambda I)^{-1}\boldsymbol{t}$$

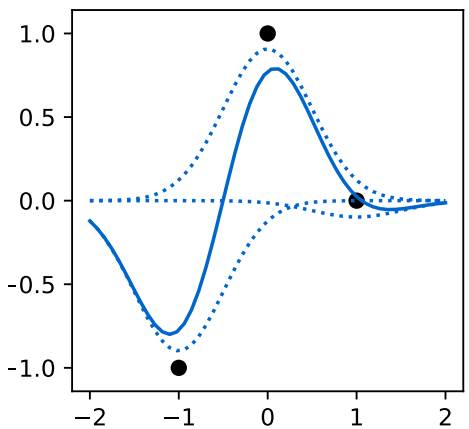
can be rewritten as a weighted sum of kernel basis functions

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} k(\boldsymbol{x}, \boldsymbol{x}_i) \cdot \alpha_i$$

where $\boldsymbol{\alpha} = (K + \lambda I)^{-1}\boldsymbol{t}$.

- The choice of the kernel function strongly influences the overall shape of the prediction function.

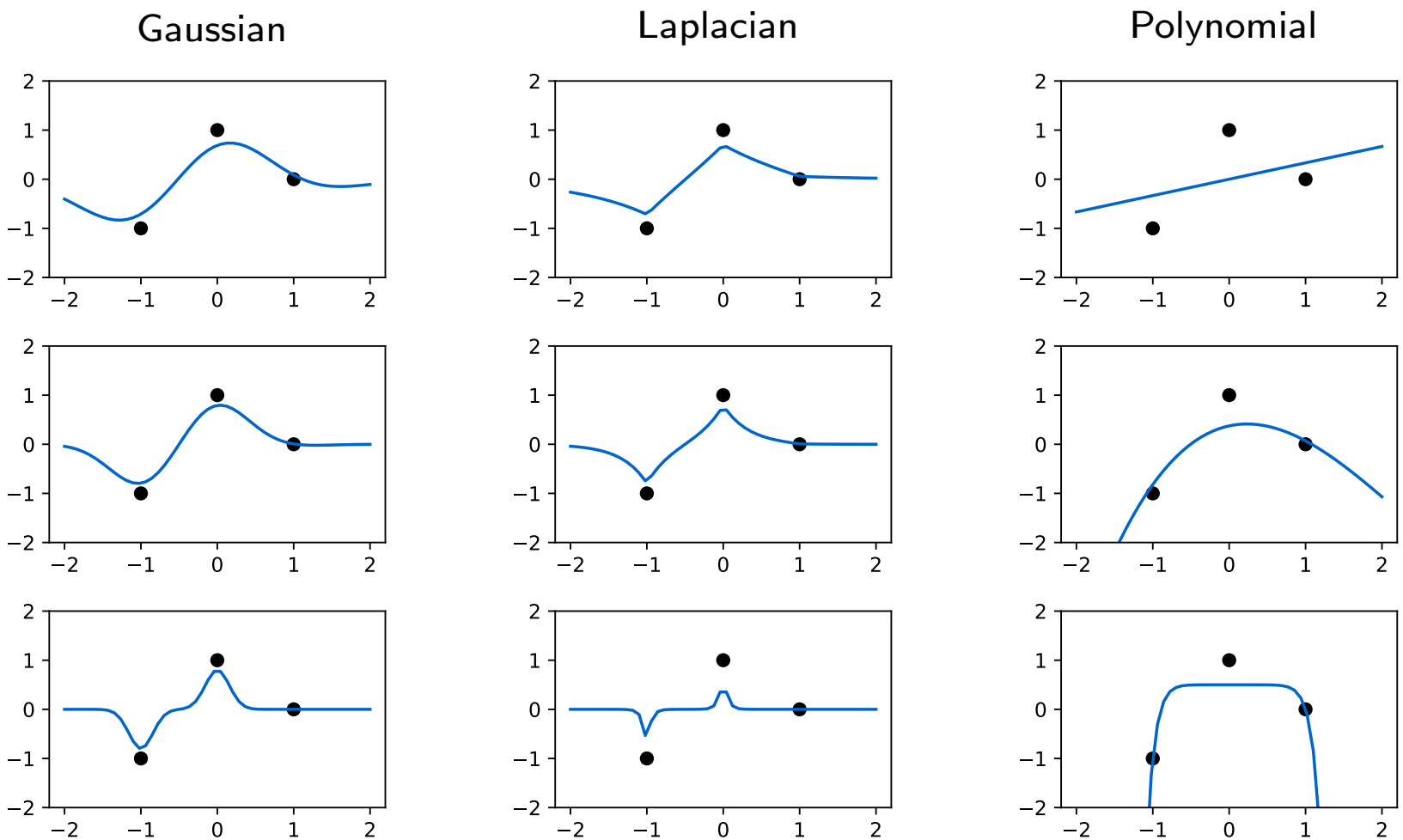
Example: Gaussian kernel ridge regression on toy one-dimensional data.



The contribution of each kernel basis function is shown as dotted lines, and the overall model is shown as a solid line.

Kernel Ridge Regression

Effect of the kernel (type of kernel and scale/degree) on the learned decision function:



Gaussian Process

- Think of regression outputs as being drawn from some joint distribution

$$p_\theta(y_1, y_2, \ldots, y_N)$$

- with the joint distribution being *Gaussian*, with covariance structure determined by the inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, i.e.

$$p_\theta(y_1, \ldots, y_N) = \mathcal{N}(0, \Sigma)$$

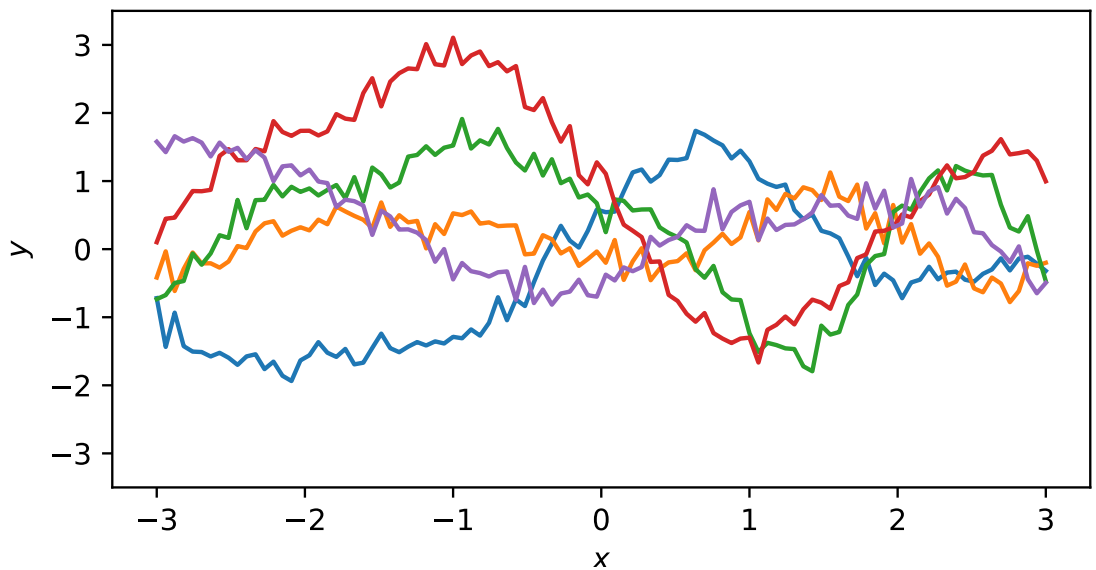
with

$$\Sigma_{ij} = k(x_i, x_j) + \sigma^2 \delta_{ij}$$

- where k can be any kernel function (e.g. Gaussian, Laplacian, polynomial, etc.) and σ^2 is the intrinsic observation noise.

Gaussian Process

Examples of 5 samples $\boldsymbol{y} \sim p_\theta(\boldsymbol{y})$ drawn from the joint distribution of targets (taking points on a grid).



- $p_\theta(\boldsymbol{y})$ can be interpreted as a 'prior' distribution on predictions.
- Targets are correlated locally in input space.
- There is some iid. noise that simulates observation noise.

Gaussian Process

- Distinguish observed data $(X, \boldsymbol{y})$ and unobserved data $(X^\star, \boldsymbol{y}^\star)$.
- They are governed by the same Gaussian distribution (now given in block matrices):

$$p_\theta\left(\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}^\star \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \Sigma_{yy} & \Sigma_{yy^\star} \\ \Sigma_{y^\star y} & \Sigma_{y^\star y^\star} \end{bmatrix}\right)$$

with

$$\Sigma_{yy} = k(X, X) + \sigma^2 I \qquad \Sigma_{yy^\star} = k(X, X^\star)$$
$$\Sigma_{y^\star y} = k(X^\star, X) \qquad \Sigma_{y^\star y^\star} = k(X^\star, X^\star) + \sigma^2 I$$

- Predict new data points by conditioning on observed targets values:

$$p_\theta(\boldsymbol{y}^\star | \boldsymbol{y})$$

- Any conditional of a Gaussian distribution is also Gaussian (of different mean and variance), these parameters can be obtained in closed form (cf. matrix cookbook).

Gaussian Process

- Using the conditional formulas for mutltivariate Gaussians, we get for a collection of future observations $y^\star$ at input locations $X^\star$ the expectation:

$$\begin{aligned}\mathbb{E}[\boldsymbol{y}^\star|\boldsymbol{y}=\boldsymbol{t}] &= \Sigma_{y^\star y}\Sigma_{yy}^{-1}\boldsymbol{t} \\ &= k(X^\star, X)(K+\sigma^2 I)^{-1}\boldsymbol{t}\end{aligned}$$

  This is the same as kernel ridge regression (with $\lambda = \sigma^2$)! Regularization parameter λ can be interpreted as a noise assumption about the labels.

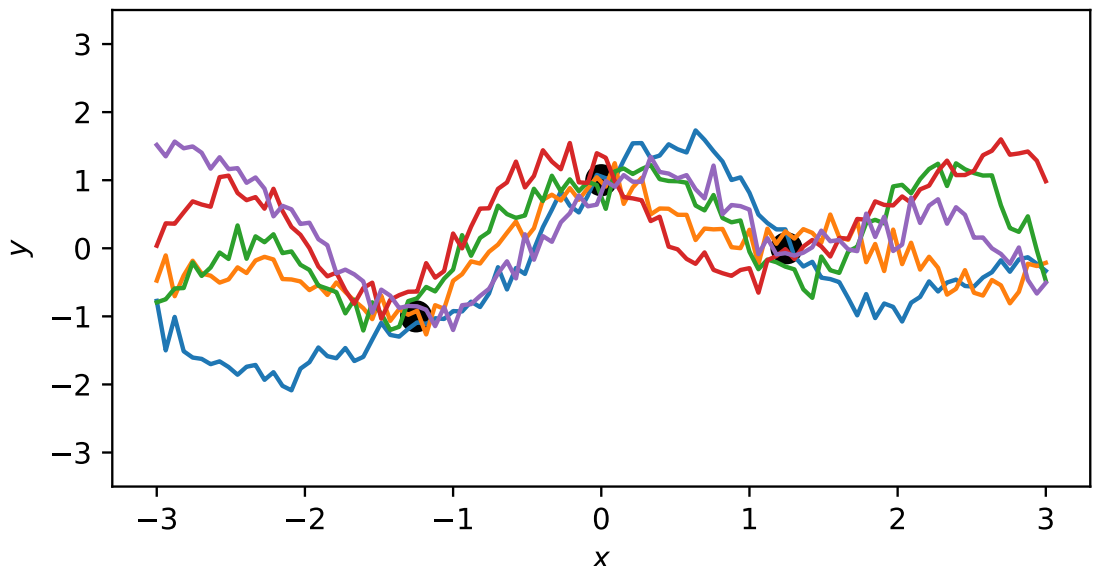
- Similarly, we can compute the covariance of the conditioned distribution

$$\begin{aligned}\mathrm{Cov}[\boldsymbol{y}^\star|\boldsymbol{y}=\boldsymbol{t}] &= \Sigma_{y^\star y^\star} - \Sigma_{y^\star y}\Sigma_{yy}^{-1}\Sigma_{yy^\star} \\ &= (k(X^\star, X^\star) + \sigma^2 I) \\ &\qquad - k(X^\star, X)(K+\sigma^2 I)^{-1}k(X, X^\star)\end{aligned}$$

  The latter provides additional information about the local uncertainty of the predictive model.

Gaussian Process

Examples of 5 samples $\boldsymbol{y}^\star \sim p_\theta(\boldsymbol{y}^\star|\boldsymbol{y})$ where we have *conditioned on some observed data* $(X, \boldsymbol{y})$ (black dots).



- Can be interpreted as a posterior distribution.
- All samples from the posterior fit the data.
- Not only expectation (best prediction), but also variance (predictive uncertainty) can be computed.

Predictive Uncertainty

Classification:

- (Kernel) logistic regression maps each data point to a class probability:

$$P(y = 1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^\top \Phi(\boldsymbol{x}))$$
$$P(y = -1|\boldsymbol{x}) = 1 - P(y = 1|\boldsymbol{x})$$

  where σ is the logistic sigmoid function.

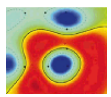
- The probability score readily indicates the confidence the model has in assigning a class to a given data point (e.g. low confidence if the probability near 0.5).

Regression:

- (Kernel) regression maps data to real-valued scores following:

$$f(\boldsymbol{x}) = \boldsymbol{w}^\top \Phi(\boldsymbol{x})$$

- No confidence measure attached to the prediction by default.
- Gaussian process enhances these models with an estimate of variance.



Robust Regression

Square loss $(y-t)^2$ is not robust to possible outliers in target values (e.g. sensor failure, human mistake in data collection).

Robust loss functions:

- Absolute loss:

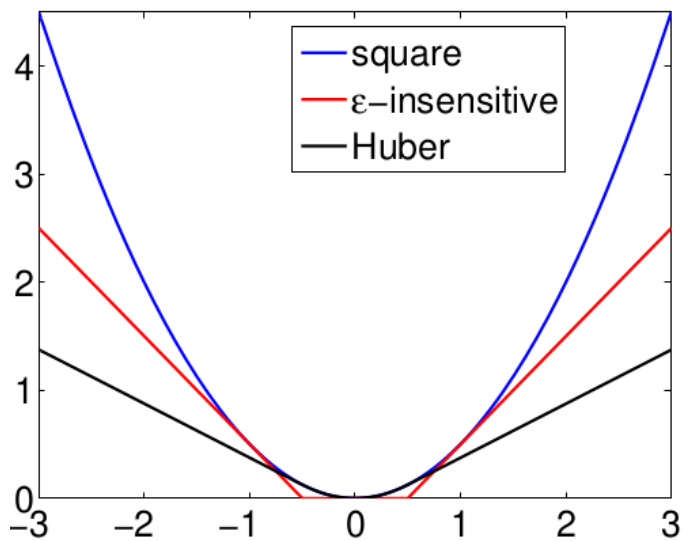
$$|y - t|.$$

- Epsilon-insensitive loss

$$\max(0, |y - t| - \varepsilon).$$

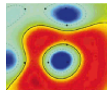
- Huber loss: $h(y - t)$ where

$$h(r) = \begin{cases} r^2 & |r| \leq c \\ 2c|r| - c^2 & \text{otherwise.} \end{cases}$$



Regression, Further Topics

- Neural networks for regression.
- Dealing with different levels of noise (heteroskedasticity)
- Enhanced error models (mixture density networks), ML2.
- Structured output learning, time series predictions, ML2.



Summary

- Regression addresses the problem of making real-valued predictions (different from classification).
- Basic regression model: least squares regression. Admits a closed form.
- Because least square regression can strongly overfit (and is undefined when $d > N$), one needs to regularize it (ridge regression).
- A nonlinear regression model can be obtained by mapping the data into some feature space and kernelizing the model and the predictions (kernel ridge regression).
- Gaussian processes extends kernel ridge regression by providing along with the prediction an estimate of variance.