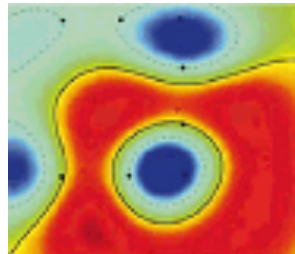
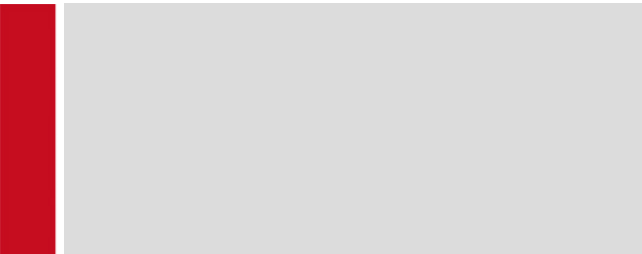




WiSe 2024/25

Machine Learning 1/1-X



## Lecture 13 | Regression

# Outline

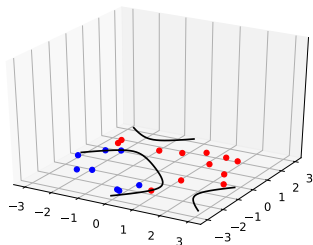
---

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

# Classification vs. Regression

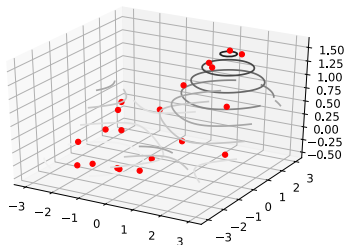
Classification

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



Regression

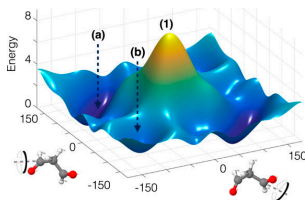
$$f : \mathbb{R}^d \rightarrow \mathbb{R}$$



# Why Regression?

- ▶ 分子动力学/化学反应的有用模型。
- ▶ 预测量（能量）本质上是实值的。
- ▶ 从能量中，可以推导出诸如作用在不同原子上的力等量。

## Example: Modeling physical systems

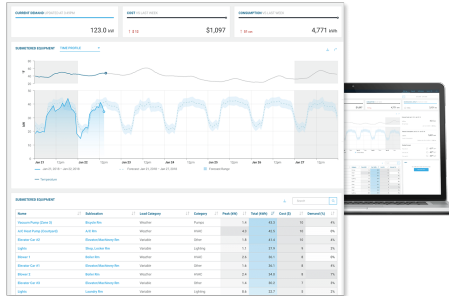


Source: Saucedo et al. J. Chem. Phys. 150, 114102 (2019)

- ▶ 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 Forecasting



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  $\omega_j$  according to the multivariate Gaussian distribution

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

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

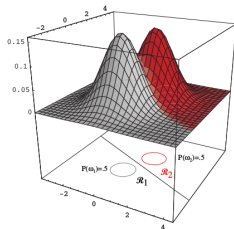
## Resulting optimal classifier:

- The Bayes optimal classifier is derived as

$$\arg \max_j \{P(\omega_j|\mathbf{x})\}$$

$$= \arg \max_j \left\{ \mathbf{x}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_j - \frac{1}{2} \boldsymbol{\mu}_j^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_j + \log P(\omega_j) \right\}$$

which is a linear classifier.



# Building an Optimal Regressor

## Example of data generating assumption:

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

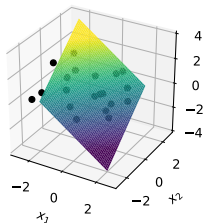
$$P(x, t) = \mathcal{N}\left(\underbrace{\begin{pmatrix} \mu_x \\ \mu_t \end{pmatrix}}_{\mu}, \underbrace{\begin{pmatrix} \Sigma_{xx} & \Sigma_{xt} \\ \Sigma_{tx} & \Sigma_{tt} \end{pmatrix}}_{\Sigma}\right)$$

## Resulting optimal regressor:

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

$$\mathbb{E}[t|x] = \mu_t + \underbrace{\Sigma_{tx}\Sigma_{xx}^{-1}}_w x + \underbrace{-\Sigma_{tx}\Sigma_{xx}^{-1}\mu_x}_b$$

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)

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

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

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

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



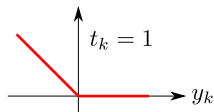
# Recap: The Perceptron



- ▶ Proposed by F. Rosenblatt in 1958.
- ▶ Classifier that perfectly 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

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



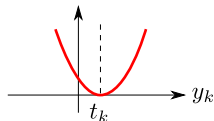
where  $t_k \in \{-1, 1\}$  and  $\max(0, -y_k t_k)$  is called the *hinge loss*.

# Hinge Loss vs. Square Loss

## Idea:

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

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



- ▶ 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:

$$\mathbf{x}^\top \mathbf{w} + b = \underbrace{[\mathbf{x}, 1]^\top}_x \underbrace{[\mathbf{w}, b]}_w$$

Therefore, the error function

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

can be rewritten more compactly as:

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

# Least Square Regression

---

The error function to minimize can be developed as:

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

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

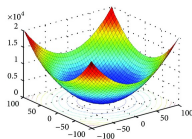
# Least Square Regression

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

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

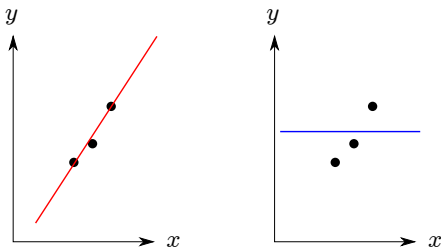
Rearranging terms and multiplying by  $(XX^\top)^{-1}$  on both sides, we get the closed form solution:

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



# 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)?

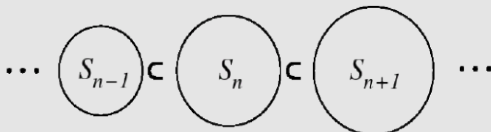
# Recap: Structural Risk Minimization (SRM)

---

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.  $\min_{\mathbf{w}} \mathcal{E}(\mathbf{w})$  s.t.  $\|\mathbf{w}\|^2 \leq C$ .
- ▶ This objective can be minimized using Lagrange multipliers:

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

which leads to:

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

where  $\lambda$  is chosen to minimize the error subject to the constraint being satisfied. This regularized least square is called *ridge regression*.

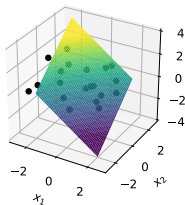
- ▶ In practice, instead of specifying the parameter  $C$ , it is common to directly treat  $\lambda$  as the hyperparameter.



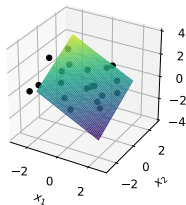
# Ridge Regression

Example:

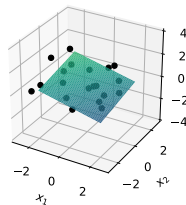
$\lambda = 0$



$\lambda = 5$



$\lambda = 25$



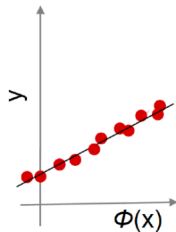
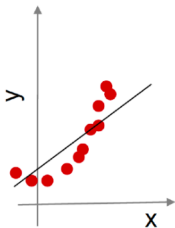
- ▶ The higher the parameter  $\lambda$ , the flatter the predicted function.
- ▶ High values of  $\lambda$  are desirable for noisy high-dimensional data.

# From Linear to NonLinear Regression

---

## Idea:

- ▶ If the function to predict is nonlinear, nonlinearly transform the input data  $x$  via some feature map  $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^h$ , 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  $y = \mathbf{w}^\top \Phi(\mathbf{x})$  where  $\mathbf{w} \in \mathbb{R}^h$  and the objective to minimize as

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

- ▶ The solution to this optimization problem is given by

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

where  $\Phi(X) = (\Phi(\mathbf{x}_1) | \dots | \Phi(\mathbf{x}_N))$ , and for an appropriate choice of parameter  $\lambda$ .

- ▶ A new data point is then predicted as

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

# Kernel Ridge Regression

---

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



we arrive at:

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

# Kernel Ridge Regression

## Observation:

- Predictions of the kernel ridge regression model:

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

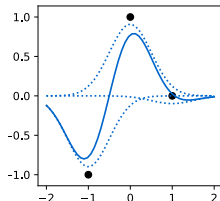
can be rewritten as a weighted sum of kernel basis functions

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

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

- 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.

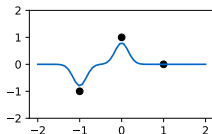
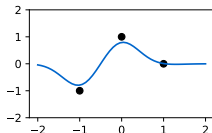
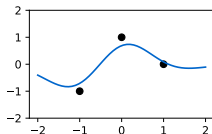


The contribution of each kernel basis function is shown as dotted lines, and the overall model is shown as a solid line.

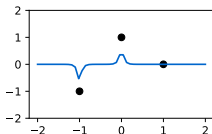
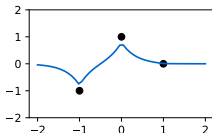
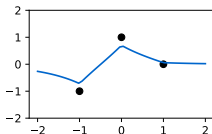
# Kernel Ridge Regression

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

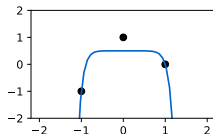
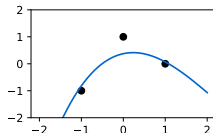
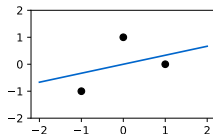
Gaussian



Laplacian



Polynomial



# Gaussian Process

---

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

$$p_{\theta}(y_1, y_2, \dots, y_N)$$

- ▶ with the joint distribution being *Gaussian*, with covariance structure determined by the inputs  $x_1, \dots, x_N$ , i.e.

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

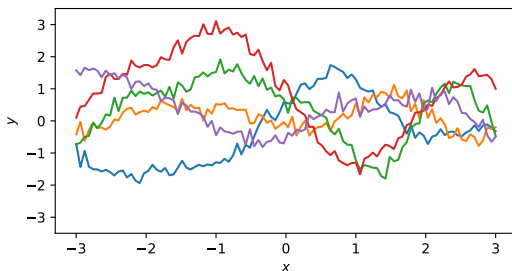
with

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

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

# Gaussian Process

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



- ▶  $p_{\theta}(\mathbf{y})$  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.

独立同分布噪声：样本曲线中存在一定的噪声，

这些噪声被假设为独立同分布的。噪声的存在模

拟了真实数据中的观测误差。



# Gaussian Process

---

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

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

with

$$\begin{aligned} \Sigma_{yy} &= k(X, X) + \sigma^2 I & \Sigma_{yy^*} &= k(X, X^*) \\ \Sigma_{y^*y} &= k(X^*, X) & \Sigma_{y^*y^*} &= k(X^*, X^*) + \sigma^2 I \end{aligned}$$

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

$$p_{\theta}(\mathbf{y}^* | \mathbf{y})$$

- ▶ 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 multivariate Gaussians, we get for a collection of future observations  $y^*$  at input locations  $X^*$  the expectation:

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

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

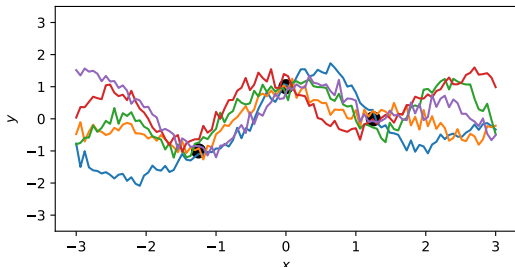
- ▶ Similarly, we can compute the covariance of the conditioned distribution

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

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

# Gaussian Process

Examples of 5 samples  $\mathbf{y}^* \sim p_{\theta}(\mathbf{y}^*|\mathbf{y})$  where we have *conditioned on some observed data*  $(X, \mathbf{y})$  (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:

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

where  $\sigma$  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:

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

- ▶ 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:

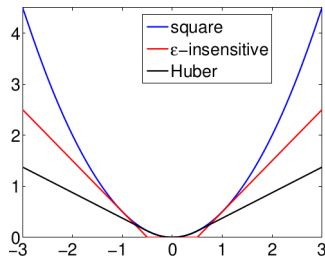
$$|y - t|.$$

- ▶ 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.

# 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.