Kernel Machines

Alain Celisse

SAMM

Paris 1-Panthéon Sorbonne University

`alain.celisse@univ-paris1.fr`

Introduction to Kernel machines

—

Master 2 Data Science – Centrale Lille, Lille University
Fall 2022

Outline of the lectures

Successive topics of the coming lectures:

1. Introduction to Kernel methods (Today!)
2. Support vector classifiers and Kernel methods
3. Extending classical strategies to high dimension
   - KRR/LS-SVMs
   - KPCA
4. Duality gap and KKT conditions
5. Designing reproducing kernels
6. Maximum Mean Discrepancy (MMD)
7. Change-point detection, KCP

Outline of the lecture

- The Big picture about kernel machines
- Focus of the regression problem
- Reproducing Kernel Hilbert Spaces (RKHSs)
- Examples of iterative learning strategies

Main challenges of modern statistical learning

The Big Picture about kernel machines

Challenges of modern statistical learning

How to overcome all of this?

Remaining difficulties/open questions

Focus on the regression problem

Reproducing Kernel Hilbert Space

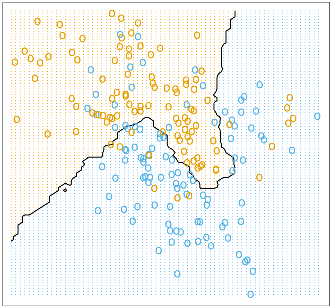
Iterative learning strategies

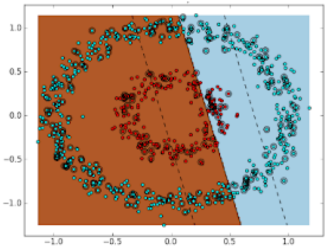
The Big Picture about kernel machines

The real world is not linear. . .

Modelizing remains difficult

- With classification/clustering tasks, classes are often overlapping
- Non linearly separable classes

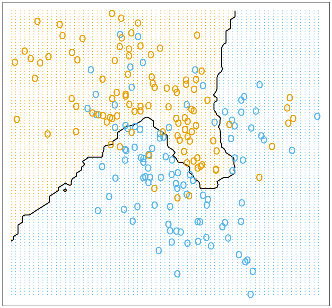


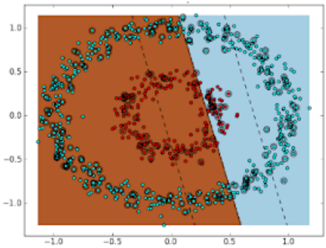


The real world is not linear...

Modelizing remains difficult

- With classification/clustering tasks, classes are often overlapping
- Non linearly separable classes





- Linear predictors: Limited performance!

Observations are complex

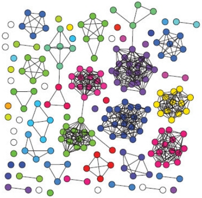
Extracting information is difficult

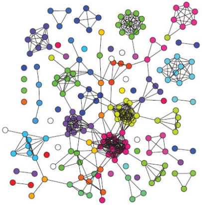
- Individuals are described by complex covariates
- Covariates may be:
  - Qualitative/ categorical: Eye color, city names, ...

Observations are complex

Extracting information is difficult

- Individuals are described by complex covariates
- Covariates may be:
  - Qualitative/ categorical: Eye color, city names, ...
  - Structured: Graphs, Images, video streams,...

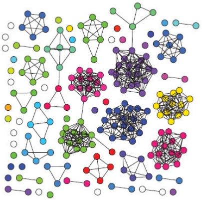


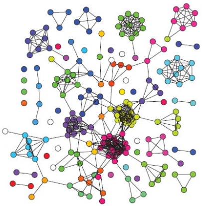


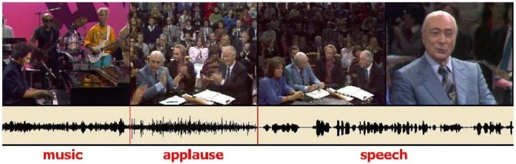
Observations are complex

Extracting information is difficult

- Individuals are described by complex covariates
- Covariates may be:
  - Qualitative/ categorical: Eye color, city names, . . .
  - Structured: Graphs, Images, video streams,. . .



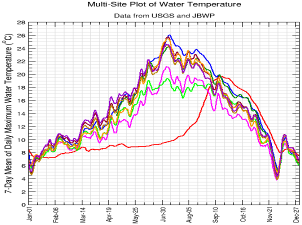




Combining structured covariates

Throwing away part of information is forbidden!

- In many applications, covariates are "heterogeneous"
- Individuals described by *mixing several types of covariates*:
    - Vectors in $\mathbb{R}^d$ (measurements)
    - Images (from social media)
    - Curves (expenses along a year)
    - ...

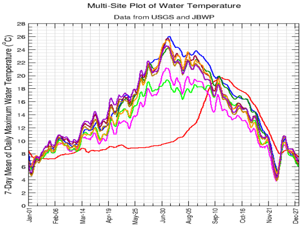


Ex: Typically used by banks to "segment" its clients

Combining structured covariates

Throwing away part of information is forbidden!

- In many applications, covariates are "heterogeneous"
- Individuals described by *mixing several types of covariates*:
  - Vectors in $\mathbb{R}^d$ (measurements)
  - Images (from social media)
  - Curves (expenses along a year)
  - ...



Ex: Typically used by banks to "segment" its clients

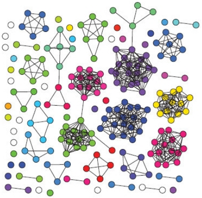
Difficult challenge

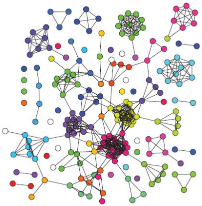
Extracting information from all covariates simulataneously

Similarity measure between “objects”

Making meaningful comparisons. . .

- Numerous strategies rely on a similarity measure (kNN, K-means, Spectral clustering,. . .)
- A similarity measure quantifies the “closeness” of points
- When points are vectors in $\mathbb{R}^d$, the Euclidean norm seems a natural choice
- When points are structured objects (graphs), there is no such natural choice!



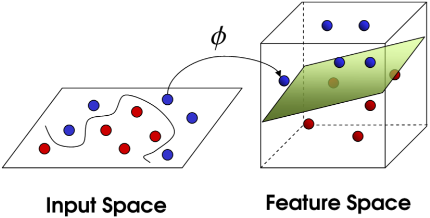


Reproducing kernels help overcoming all of this!

Beyond linearity. . .

Kernels alleviate the limitation of linear classifiers

- "Kernels" are tools outperforming linear classifiers (SVM)



- Original observations X_is are mapped into a "Feature space" of higher dimension
- The "new observations" $Y_i = \phi(X_i)$s are vectors (The Feature space is a vector space!)
- The Input space not necessarily a Vector space

Capturing features of the probability distribution

General principle

- No longer look for changes among $X_1, \ldots, X_n$: Forget the Input space
- Rather look for changes among the new observations $Y_1, \ldots, Y_n$ within the Feature space!

Capturing features of the probability distribution

The Big Picture about kernel machines
Challenges of modern statistical learning
How to overcome all of this?
Remaining difficulties/open questions
Focus on the regression problem
Reproducing Kernel Hilbert Space
Iterative learning strategies

General principle

- No longer look for changes among $X_1, \ldots, X_n$: Forget the Input space
- Rather look for changes among the new observations $Y_1, \ldots, Y_n$ within the Feature space!

Assets

- Detect changes between the probability distributions of the X_is: $P_{X_1}, \ldots, P_{X_n}$ (see Mean embedding, MMD)

Capturing features of the probability distribution

General principle

- No longer look for changes among $X_1, \ldots, X_n$: Forget the Input space
- Rather look for changes among the new observations $Y_1, \ldots, Y_n$ within the Feature space!

Assets

- Detect changes between the probability distributions of the X_is: $P_{X_1}, \ldots, P_{X_n}$ (see Mean embedding, MMD)
- Yields new measures of dependence between the X_i:s No longer limited to covariance and linear dependence (see HSIC)

Kernels on "objects"

Kernels are a versatile tool

Defined for various types of objects:

(Kernel for structured data (2008), T. Gärtner)

- Vectors

$$k(a, b) = e^{-\frac{(a-b)^2}{2h}}, \qquad a, b \in \mathbb{R}$$

- Sets/Measurable sets

$$k(A, B) = \mu\left(A \cap B\right), \qquad A, B \in \mathcal{P}\left(\mathbb{R}\right)$$

- Histograms, Graphs, Curves,...

Kernels on "objects"

Kernels are a versatile tool

Defined for various types of objects:

(Kernel for structured data (2008), T. Gärtner)

- Vectors

$$k(a, b) = e^{-\frac{(a-b)^2}{2h}}, \qquad a, b \in \mathbb{R}$$

- Sets/Measurable sets

$$k(A, B) = \mu(A \cap B), \qquad A, B \in \mathcal{P}(\mathbb{R})$$

- Histograms, Graphs, Curves,...

Designing new kernels

Simple mathematical rules allow for building new kernels:

- Sum
- Product
- Convex combination, ...

Combining covariates is easy

Dealing with marginal information

Assume

$$X_i = (X_i^1, X_i^2, \dots, X_i^p)$$

with covariates

- $X_i^1 \in \mathcal{X}_1 = \mathbb{R}^d$: Measurements $\rightarrow \quad k_1(\cdot, \cdot)$
- $X_i^2 \in \mathcal{X}_2$: Curves on $[0, 1] \rightarrow \quad k_2(\cdot, \cdot)$
- $X^3 \in \mathcal{X}_3$: Medical images of a patient $\rightarrow \quad k_3(\cdot, \cdot)$

Combining covariates is easy

The Big Picture about kernel machines

Challenges of modern statistical learning

How to overcome all of this?

Remaining difficulties/open questions

Focus on the regression problem

Reproducing Kernel Hilbert Space

Iterative learning strategies

Dealing with marginal information

Assume

$$X_i = (X_i^1, X_i^2, \ldots, X_i^p)$$

with covariates

- $X_i^1 \in \mathcal{X}_1 = \mathbb{R}^d$: Measurements $\rightarrow \quad k_1(\cdot,\cdot)$
- $X_i^2 \in \mathcal{X}_2$: Curves on $[0,1] \rightarrow \quad k_2(\cdot,\cdot)$
- $X^3 \in \mathcal{X}_3$: Medical images of a patient $\rightarrow \quad k_3(\cdot,\cdot)$

Gathering all these complementary information sources

$$k(X_i, X_j) = \sum_{\ell=1}^{p} \omega_\ell k_\ell(X_i^\ell, X_j^\ell), \qquad \omega_\ell \geq 0$$

$\rightarrow$ Individuals i and j are compared by means of all covariates

Remaining difficulties/open questions

Optimizing the kernel/metric . . .

- The practical performance depends on the kernel
  - Gaussian kernel: $k(a,b) = e^{-\frac{(a-b)^2}{2h}}$
  - Laplace kernel: $k(a,b) = e^{-\frac{|a-b|}{h}}$

$\rightarrow$ A bad choice leads to poor performances

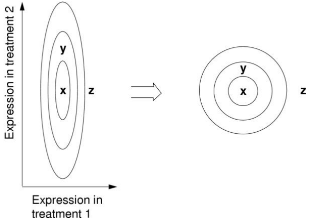
Optimizing the kernel/metric ...

- The practical performance depends on the kernel
  - Gaussian kernel: $k(a,b) = e^{-\frac{(a-b)^2}{2h}}$
  - Laplace kernel: $k(a,b) = e^{-\frac{|a-b|}{h}}$

  $\rightarrow$ A bad choice leads to poor performances

- Same problem as with the choice of the metric

  $\rightarrow$ Reweighting covariates ...



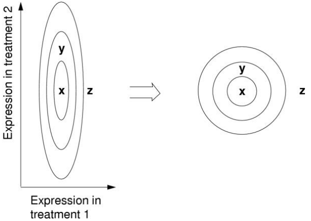
Optimizing the kernel/metric . . .

- The practical performance depends on the kernel
  - Gaussian kernel: $k(a,b) = e^{-\frac{(a-b)^2}{2h}}$
  - Laplace kernel: $k(a,b) = e^{-\frac{|a-b|}{h}}$

$\rightarrow$ A bad choice leads to poor performances

- Same problem as with the choice of the metric

$\rightarrow$ Reweighting covariates . . .



- A "Kernel" refers to a parametric family of functions

$\rightarrow$ Gaussian kernel parametrized by $h > 0$

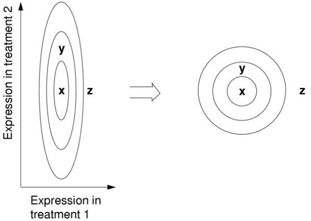
Optimizing the kernel/metric . . .

- The practical performance depends on the kernel
  - Gaussian kernel: $k(a,b) = e^{-\frac{(a-b)^2}{2h}}$
  - Laplace kernel: $k(a,b) = e^{-\frac{|a-b|}{h}}$

  $\rightarrow$ A bad choice leads to poor performances

- Same problem as with the choice of the metric

  $\rightarrow$ Reweighting covariates . . .



- A "Kernel" refers to a parametric family of functions

  $\rightarrow$ Gaussian kernel parametrized by $h > 0$
- Challenge: Optimizing the kernel remains widely open!

Computational challenges

The so-called Gram matrix

- A kernel $(a, b) \mapsto k(a, b)$
- From $X_1, \ldots, X_n$, compute the Gram matrix $K = \{K_{i,j}\}_{1 \leq i,j \leq n}$, where

$$K_{i,j} = k(X_i, X_j)$$

Computational challenges

The so-called Gram matrix

- A kernel $(a, b) \mapsto k(a, b)$
- From $X_1, \ldots, X_n$, compute the Gram matrix $K = \{K_{i,j}\}_{1 \leq i,j \leq n}$, where

$$K_{i,j} = k(X_i, X_j)$$

- Gram matrix K: $n \times n$ matrix
- Most of kernel machines rely on computing K

Computational challenges

The so-called Gram matrix

- A kernel $(a, b) \mapsto k(a, b)$
- From $X_1, \ldots, X_n$, compute the Gram matrix $K = \{K_{i,j}\}_{1 \leq i,j \leq n}$, where

$$K_{i,j} = k(X_i, X_j)$$

- Gram matrix K: $n \times n$ matrix
- Most of kernel machines rely on computing K

Computational issues

- Computing K: $O(n^2)$ time-complexity

Computational challenges

The so-called Gram matrix

- A kernel $(a, b) \mapsto k(a, b)$
- From $X_1, \ldots, X_n$, compute the Gram matrix $K = \{K_{i,j}\}_{1 \leq i,j \leq n}$, where

$$K_{i,j} = k(X_i, X_j)$$

- Gram matrix K: $n \times n$ matrix
- Most of kernel machines rely on computing K

Computational issues

- Computing K: $O(n^2)$ time-complexity
- Storing K: $O(n^2)$ space-complexity

Computational challenges

The so-called Gram matrix

- A kernel $(a, b) \mapsto k(a, b)$
- From $X_1, \ldots, X_n$, compute the Gram matrix $K = \{K_{i,j}\}_{1 \le i,j \le n}$, where

$$K_{i,j} = k(X_i, X_j)$$

- Gram matrix K: $n \times n$ matrix
- Most of kernel machines rely on computing K

Computational issues

- Computing K: $O(n^2)$ time-complexity
- Storing K: $O(n^2)$ space-complexity

Remarks:

- Requires cautious computations
- Approximation techniques to speed up computations

Focus on the regression problem

Regression task

The Big Picture about kernel machines

Focus on the regression problem

Regression task

Review of tentative solutions

Finally kernels come into play. . .

Reproducing Kernel Hilbert Space

Iterative learning strategies

The probabilistic model

Regression Model

For $X \in \mathbb{R}^d$,

$$Y = f^\star(X) + \epsilon \in \mathbb{R}$$

Assumptions:

- $f^\star(x) = \mathbb{E}\left[\, Y \mid X = x \,\right]$ (regression function)
- $\mathbb{E}\left[\, \epsilon \mid X = x \,\right] = 0$
- $\mathrm{Var}\left[\, \epsilon \mid X = x \,\right] \leq \sigma^2 < +\infty$

Remark:

Estimating $f^\star$ amounts to learning the link between X and Y

Cost function, loss and risk

- Quadratic cost function:

$$c\left(f(x), y\right) = \left(f(x) - y\right)^2$$

Cost function, loss and risk

- Quadratic cost function:

$$c\left(f(x), y\right) = \left(f(x) - y\right)^2$$

- Prediction error (Loss):

$$\begin{aligned} PE(f) &= \mathbb{E}_{(X,Y)\sim P}\left[c\left(f(X), Y\right)\right] \\ &= \mathbb{E}_{(X,Y)\sim P}\left[\left(f(X) - Y\right)^2\right] \end{aligned}$$

Remark: Note that

$$PE(f^\star) = \inf_{h \in \mathcal{M}(\mathbb{R}^d)} PE(h)$$

Cost function, loss and risk

- Quadratic cost function:

$$c\left(f(x), y\right) = \left(f(x) - y\right)^2$$

- Prediction error (Loss):

$$\begin{aligned} PE(f) &= \mathbb{E}_{(X,Y)\sim P}\left[c\left(f(X), Y\right)\right] \\ &= \mathbb{E}_{(X,Y)\sim P}\left[\left(f(X) - Y\right)^2\right] \end{aligned}$$

Remark: Note that

$$PE(f^\star) = \inf_{h \in \mathcal{M}(\mathbb{R}^d)} PE(h)$$

- Excess Loss:

$$\begin{aligned} \mathcal{E}(f) &= PE(f) - \inf_{h \in \mathcal{M}} PE(h) \\ &= \mathbb{E}_{X \sim P_X}\left[\left(f(X) - f^\star(X)\right)^2\right] \\ &= \|f - f^\star\|^2_{L^2(P_X)} \end{aligned}$$

The functional perspective

Statistical model

$$Y = f(X) + \epsilon \in \mathbb{R}, \qquad \text{with} \quad f \in \mathcal{F}$$

- $\mathcal{F}$: set of candidate functions
- The best estimator of $f^\star$ within $\mathcal{F}$:

$$f^\star_{\mathcal{F}} = Arg \min_{f \in \mathcal{F}} PE(f)$$

The functional perspective

Statistical model

$$Y = f(X) + \epsilon \in \mathbb{R}, \qquad \text{with} \quad f \in \mathcal{F}$$

- $\mathcal{F}$: set of candidate functions
- The best estimator of $f^\star$ within $\mathcal{F}$:

$$f^\star_{\mathcal{F}} = Arg \min_{f \in \mathcal{F}} PE(f)$$

Bias-Variance trade-off

For any estimator $\widehat{f}$ of $f^\star$

$$\begin{aligned}\mathcal{E}(\widehat{f}) &= PE(\widehat{f}) - PE(f^\star) \\ &= \underbrace{PE(\widehat{f}) - \inf_{f \in \mathcal{F}} PE(f)}_{=\text{Variance term}} + \underbrace{\inf_{f \in \mathcal{F}} PE(f) - PE(f^\star)}_{=\text{Bias term}}\end{aligned}$$

Review of tentative solutions

Kernel Machines

Alain Celisse

The Big Picture about kernel machines

Focus on the regression problem

Regression task

Review of tentative solutions

Finally kernels come into play. . .

Reproducing Kernel Hilbert Space

Iterative learning strategies

Linear regression model

The Big Picture about kernel machines

Focus on the regression problem

Regression task

Review of tentative solutions

Finally kernels come into play. . .

Reproducing Kernel Hilbert Space

Iterative learning strategies

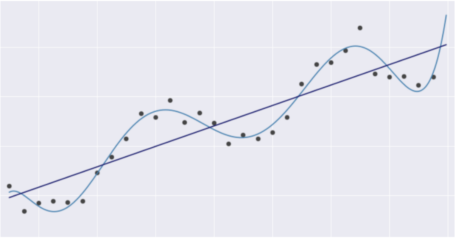
$$\mathcal{F} = \left\{ x \mapsto f(x) = \langle x, \beta \rangle_{\mathbb{R}^d} \mid \beta \in \mathbb{R}^d \right\}$$

- Best approximation to $f^\star$:

$$f^\star_{\mathcal{F}} = \langle \cdot, \beta^\star \rangle_{\mathbb{R}^d}$$

- The linear regression model is likely not the true one!
- This means that the bias satisfies

$$\inf_{f \in \mathcal{F}} PE(f) - PE(f^\star) > 0$$



Single and multi-index models

$$\mathcal{F} = \{f(x) = g(Bx) \mid B \in \mathcal{M}_{m,d}(\mathbb{R}),\ g \text{ non-linear}\}$$

- If $m = 1$, Single-Index Model (SIM)
- If $1 < m \leq d$, Multi-Index Model (MIM)

Single and multi-index models

$$\mathcal{F} = \{f(x) = g(Bx) \mid B \in \mathcal{M}_{m,d}(\mathbb{R}),\ g \text{ non-linear}\}$$

- If $m = 1$, Single-Index Model (SIM)
- If $1 < m \leq d$, Multi-Index Model (MIM)

Difficulties

- The non-linear function $g : \mathbb{R}^m \to \mathbb{R}$ is unknown
- Since both g and B are unknown, difficult to estimate
- Often monotonicity assumptions added on g to make problem easier

Deep Neural Networks (DNNs)

Activation function

- σ: activation function
- **Ex:** $\sigma(u) = \max\{0, u\}$ (ReLU)

Single-hidden layer DNN

$$\mathcal{F} = \left\{ f(x) = \underbrace{\sigma(B^1 x + c^1)}_{=\phi_1(x)} \mid B^1 \in \mathcal{M}_{m,d}(\mathbb{R}),\ c^1 \in \mathbb{R}^m \right\}$$

Deep Neural Networks (DNNs)

Activation function

- σ: activation function
- **Ex:** $\sigma(u) = \max\{0, u\}$ (ReLU)

Single-hidden layer DNN

$$\mathcal{F} = \left\{ f(x) = \underbrace{\sigma(B^1 x + c^1)}_{=\phi_1(x)} \mid B^1 \in \mathcal{M}_{m,d}(\mathbb{R}),\ c^1 \in \mathbb{R}^m \right\}$$

Multi-layer DNN (MLP)

$$\mathcal{F} = \{ f(x) = \phi_N \circ \phi_{N-1} \circ \cdots \circ \phi_1(x) \}$$

where $\phi_j(u) = \sigma\left(B^j u + c^j\right)$ for all $1 \leq j \leq N$

Deep Neural Networks (DNNs)

Activation function

- σ: activation function
- **Ex:** $\sigma(u) = \max\{0, u\}$ (ReLU)

Single-hidden layer DNN

$$\mathcal{F} = \left\{ f(x) = \underbrace{\sigma(B^1 x + c^1)}_{=\phi_1(x)} \mid B^1 \in \mathcal{M}_{m,d}(\mathbb{R}),\ c^1 \in \mathbb{R}^m \right\}$$

Multi-layer DNN (MLP)

$$\mathcal{F} = \{ f(x) = \phi_N \circ \phi_{N-1} \circ \cdots \circ \phi_1(x) \}$$

where $\phi_j(u) = \sigma\left(B^j u + c^j\right)$ for all $1 \leq j \leq N$

Problems:

- Not convex: Many local optima
- Difficult to understand

Finally kernels come into play. . .

Kernel Machines

Alain Celisse

The Big Picture about kernel machines

Focus on the regression problem

Regression task

Review of tentative solutions

Finally kernels come into play. . .

Reproducing Kernel Hilbert Space

Iterative learning strategies

Towards the RKHS

$\mathcal{F}$ can be chosen to be a Hilbert space with specific properties called Reproducing Kernel Hilbert Space (RKHS)

Definition (RKHS)

A Hilbert space $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ is an RKHS if there exists a map $k : \mathcal{X}^2 \to \mathbb{R}$ such that

- $x \mapsto k_x = k(x, \cdot) \in \mathcal{H}$
- For all $g \in \mathcal{H}$,

$$h(x) = \langle h, k_x \rangle_{\mathcal{H}} , \qquad \forall x \in \mathcal{X}$$

Then, k is called a reproducing kernel

□

Ex: $\mathcal{X} = \mathbb{R}$, $k(x, y) = \langle x, y \rangle_{\mathbb{R}^d}$.
Then $\mathcal{H} = \left\{ x \mapsto f_\beta(x) = \langle \beta, x \rangle_{\mathbb{R}^d} \mid \beta \in \mathbb{R}^d \right\}$ is an RKHS

Reproducing Kernel Hilbert Space (RKHS)

From RKHS to reproducing kernels

Mercer's theorem

Set $T_k : L^2(\rho) \to L^2(\rho)$ a linear operator

$$x \mapsto T_k(f)(x) = \int_{\mathcal{X}} k(x,u) f(u) d\rho(u), \qquad \forall f \in \mathcal{H}$$

Theorem (Mercer's theorem)

If T_k is compact self-adjoint, then there exist:

- *an orthonormal family $\{\psi_\ell\}_{\ell \geq 1}$ of eigenfunctions of T_k,*
- *a non-increasing sequence $\lambda_1 \geq \cdots \geq \lambda_n \geq \cdots \geq 0$ of eigenvalues of T_k such that*

$$k(x,y) = \sum_{\ell \geq 1} \lambda_\ell \psi_\ell(x) \psi_\ell(y) = \langle \phi(x), \phi(y) \rangle_{\ell^2}$$

with $\phi(x) = \left\{ \sqrt{\lambda_\ell} \psi_\ell(x) \right\}_{\ell \geq 1} \in \ell^2(\mathbb{R})$

Describing the RKHS

Theorem

Let

$H = \left\{ f \in L^2(\rho) \mid f = \sum_{\ell \geq 1} \theta_\ell \psi_\ell, \text{ and } \sum_{\ell \geq 1} \frac{\theta_\ell^2}{\lambda_\ell} < +\infty \right\}$

and

$$\langle f, g \rangle_{\mathcal{H}} = \sum_{\ell \geq 1} \frac{\theta_\ell^f \theta_\ell^g}{\lambda_\ell}$$

with

- $f = \sum_{\ell \geq 1} \theta_\ell^f \psi_\ell$
- $g = \sum_{\ell \geq 1} \theta_\ell^g \psi_\ell$

Then, $\mathcal{H}$ is the RKHS associated with k □

Remarks:

- $\mathcal{H}$ is a space of functions
- Smoothness encoded by the decay rate of the λ_ℓs
- The faster the λ_ℓs to 0, the smoother the functions in $\mathcal{H}$

Examples of Reproducing Kernels

Classical examples

- Linear kernel:

$$k(x, y) = \langle x, y \rangle_{\mathbb{R}^d}$$

- Polynomial kernel: $(c \geq 0,\ d > 0)$

$$k(x, y) = (\langle x, y \rangle_{\mathbb{R}^d} + c)^d$$

- Gaussian (Radial Basis Function) kernel:

$$k(x, y) = e^{-\frac{(x-y)^2}{2}}$$

- Exponential kernel:

$$k(x, y) = e^{\langle x, y \rangle_{\mathbb{R}^d}}$$

First examples of iterative learning strategies

The Gradient Descent algorithm

Empirical risk

From training data

$$\mathrm{Y} = F^\star + \boldsymbol{\epsilon} \in \mathbb{R}^n$$

where

- $\mathrm{Y} = (Y_1, \ldots, Y_n)^\top$,
- $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^\top$,
- $F^\star = (f^\star(X_1), \ldots, f^\star(X_n))^\top$

Empirical risk

$$PE(f) \quad \approx \quad \widehat{R}(f) = \frac{1}{n} \|\mathrm{Y} - F\|_2^2 = \|\mathrm{Y} - F\|_n^2$$

with $F = (f(X_1), \ldots, f(X_n))^\top$

Empirical risk

From training data

$$\mathrm{Y} = F^\star + \boldsymbol{\epsilon} \in \mathbb{R}^n$$

where

- $\mathrm{Y} = (Y_1, \ldots, Y_n)^\top$,
- $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)^\top$,
- $F^\star = (f^\star(X_1), \ldots, f^\star(X_n))^\top$

Empirical risk

$$PE(f) \quad \approx \quad \widehat{R}(f) = \frac{1}{n} \|\mathrm{Y} - F\|_2^2 = \|\mathrm{Y} - F\|_n^2$$

with $F = (f(X_1), \ldots, f(X_n))^\top$

Question

What if we were minimizing $\widehat{R}(f)$ over $\mathcal{H}$?

Representer theorem and overfitting (1/2)

Theorem (Representer theorem)

$\Psi : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}$, nondecreasing w.r.t. its $n+1$th argument

$$Arg \min_{g \in \mathcal{H}} \left\{ \Psi \left[g(x_1), \ldots, g(x_n), \|g\|_{\mathcal{H}} \right] \right\}$$

Any solution $\widehat{g}$ to the above optimization problem can be written as

$$\widehat{g}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i k(x_i, x), \quad \forall x \in \mathcal{X}$$

where $\widehat{\alpha}_i \in \mathbb{R}$, for all $1 \leq i \leq n$

Representer theorem and overfitting (1/2)

Theorem (Representer theorem)

$\Psi : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}$, *nondecreasing w.r.t. its* $n+1$*th argument*

$$Arg \min_{g \in \mathcal{H}} \left\{ \Psi \left[g(x_1), \ldots, g(x_n), \|g\|_{\mathcal{H}} \right] \right\}$$

Any solution $\widehat{g}$ *to the above optimization problem can be written as*

$$\widehat{g}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i k(x_i, x), \quad \forall x \in \mathcal{X}$$

where $\widehat{\alpha}_i \in \mathbb{R}$*, for all* $1 \leq i \leq n$

Application:

Minimizing the empirical risk $\widehat{R}(f) = \|Y - F\|_n^2$ over $\mathcal{H}$ ...

Representer theorem and overfitting (2/2)

Applying the representer theorem ... (K: Gram matrix)

$$
\begin{aligned}
Arg \min_{f \in \mathcal{H}} \widehat{R}(f) &= Arg \min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - f(x_i))^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{n} \alpha_j k(x_j, x_i) \right)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - [K\alpha]_i)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \|\mathrm{Y} - K\alpha\|_n^2 \right\}
\end{aligned}
$$

Representer theorem and overfitting (2/2)

Applying the representer theorem . . . (K: Gram matrix)

$$
\begin{aligned}
Arg \min_{f \in \mathcal{H}} \widehat{R}(f) &= Arg \min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - f(x_i))^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{n} \alpha_j k(x_j, x_i) \right)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - [K\alpha]_i)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \|\mathrm{Y} - K\alpha\|_n^2 \right\}
\end{aligned}
$$

- K: full rank $\longrightarrow$ unique solution $\widehat{\alpha} \in \mathbb{R}^n$ and $\widehat{R}(\widehat{f}) = 0$!
- K: finite rank $\longrightarrow$ many solutions (but $\widehat{R}(\widehat{f}) \neq 0$)

Representer theorem and overfitting (2/2)

Applying the representer theorem ... (K: Gram matrix)

$$
\begin{aligned}
Arg \min_{f \in \mathcal{H}} \widehat{R}(f) &= Arg \min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - f(x_i))^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{n} \alpha_j k(x_j, x_i) \right)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - [K\alpha]_i)^2 \right\} \\
&= Arg \min_{\alpha \in \mathbb{R}^n} \left\{ \|Y - K\alpha\|_n^2 \right\}
\end{aligned}
$$

- K: full rank $\longrightarrow$ unique solution $\widehat{\alpha} \in \mathbb{R}^n$ and $\widehat{R}(\widehat{f}) = 0$!
- K: finite rank $\longrightarrow$ many solutions (but $\widehat{R}(\widehat{f}) \neq 0$)

Strategy:

Constrain the solutions to avoid overfitting!

Gradient descent (GD)

Empirical risk

$$(F = (f(X_1), \ldots, f(X_n))^\top)$$

$$\widehat{R}(f) = \|\mathrm{Y} - F\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \langle f, k_{X_i}\rangle_{\mathcal{H}}\right)^2$$

Gradient descent (GD)

Empirical risk $\quad (F = (f(X_1), \ldots, f(X_n))^\top)$

$$\widehat{R}(f) = \|\mathrm{Y} - F\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \langle f, k_{X_i}\rangle_{\mathcal{H}}\right)^2$$

First order approx.

$$\widehat{R}(f) \approx \widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

Gradient descent (GD)

Empirical risk $\quad (F = (f(X_1), \ldots, f(X_n))^\top)$

$$\widehat{R}(f) = \|\mathrm{Y} - F\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \langle f, k_{X_i}\rangle_{\mathcal{H}}\right)^2$$

First order approx.

$$\widehat{R}(f) \approx \widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

$\rightarrow$ Minimizing the above expression w.r.t. $f \in \mathcal{H}$ yields

$$f - f^t \propto_{>0} -\frac{\nabla_{f^t}\widehat{R}}{\left\|\nabla_{f^t}\widehat{R}\right\|_{\mathcal{H}}} \quad \in \mathcal{H}$$

Gradient descent (GD)

Empirical risk $(F = (f(X_1), \ldots, f(X_n))^\top)$

$$\widehat{R}(f) = \|\mathrm{Y} - F\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \langle f, k_{X_i}\rangle_{\mathcal{H}}\right)^2$$

First order approx.

$$\widehat{R}(f) \approx \widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

$\rightarrow$ Minimizing the above expression w.r.t. $f \in \mathcal{H}$ yields

$$f - f^t \propto_{>0} -\frac{\nabla_{f^t}\widehat{R}}{\left\|\nabla_{f^t}\widehat{R}\right\|_{\mathcal{H}}} \quad \in \mathcal{H}$$

Gradient descent updates

For $0 < \alpha$ (small),

$$f^0 = 0$$

$$f^{t+1} = f^t - \frac{\alpha}{2}\nabla_{f^t}\widehat{R} \quad \in \mathcal{H}$$

Sketch of proof

First step Minimizing $\widehat{R}(f)$ w.r.t. f amounts to minimizing

$$\widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}} = \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

Sketch of proof

The Big Picture about kernel machines

Focus on the regression problem

Reproducing Kernel Hilbert Space

Iterative learning strategies

Gradient descent algorithm

Stochastic Gradient Descent algorithm

First step Minimizing $\widehat{R}(f)$ w.r.t. f amounts to minimizing

$$\widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}} = \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

Second step With $f = f^t + \delta g \in \mathcal{H}$ ($\|g\|_{\mathcal{H}} = 1$, $\delta > 0$), it amounts to minimize

$$\left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}} = \delta \left\langle \nabla_{f^t}\widehat{R}, g \right\rangle_{\mathcal{H}} \geq -\delta \left\| \nabla_{f^t}\widehat{R} \right\|_{\mathcal{H}} \cdot \|g\|_{\mathcal{H}}$$

Sketch of proof

The Big Picture about kernel machines

Focus on the regression problem

Reproducing Kernel Hilbert Space

Iterative learning strategies

Gradient descent algorithm

Stochastic Gradient Descent algorithm

First step Minimizing $\widehat{R}(f)$ w.r.t. f amounts to minimizing

$$\widehat{R}(f^t) + \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}} = \left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}}$$

Second step With $f = f^t + \delta g \in \mathcal{H}$ ($\|g\|_{\mathcal{H}} = 1$, $\delta > 0$), it amounts to minimize

$$\left\langle \nabla_{f^t}\widehat{R}, f - f^t \right\rangle_{\mathcal{H}} = \delta \left\langle \nabla_{f^t}\widehat{R}, g \right\rangle_{\mathcal{H}} \geq -\delta \left\| \nabla_{f^t}\widehat{R} \right\|_{\mathcal{H}} \cdot \|g\|_{\mathcal{H}}$$

Third step Achieved at $g = -\nabla_{f^t}\widehat{R} / \left\| \nabla_{f^t}\widehat{R} \right\|_{\mathcal{H}}$

$$\longrightarrow \qquad f^{t+1} = f^t - \delta \nabla_{f^t}\widehat{R} / \left\| \nabla_{f^t}\widehat{R} \right\|_{\mathcal{H}} = f^t - \frac{\alpha}{2}\nabla_{f^t}\widehat{R}$$

for a well-chosen step size $\alpha > 0$ (which can depend on t)

GD estimator: Closed-form expression

For $0 < \alpha$,

$$f^0 = 0$$
$$f^{t+1} = f^t - \frac{\alpha}{2} \nabla_{f^t} \widehat{R} \quad \in \mathcal{H}$$

Closed-form expression:

$$F^t = (f^t(X_1), \ldots, f^t(X_n)) \in \mathbb{R}^n$$

GD estimator: Closed-form expression

For $0 < \alpha$,

$$f^0 = 0$$

$$f^{t+1} = f^t - \frac{\alpha}{2} \nabla_{f^t} \widehat{R} \quad \in \mathcal{H}$$

Closed-form expression:

$$F^t = (f^t(X_1), \ldots, f^t(X_n)) \in \mathbb{R}^n$$

$$\begin{cases} F^t & = \left[I_n - \Pi_{s=1}^t (I_n - \alpha K_n) \right] Y, \qquad t \geq 1 \\ F^0 & = 0 \end{cases}$$

with

- $K_n = K/n$: normalized Gram matrix
- $\hat{\mu}_1 \geq \cdots \geq \hat{\mu}_n \geq 0$: nonincreasing eigenvalues of K_n
- α such that $\alpha \hat{\mu}_1 < 1 \to$ Why?

GD estimator: Closed-form expression

For $0 < \alpha$,

$$f^0 = 0$$
$$f^{t+1} = f^t - \frac{\alpha}{2} \nabla_{f^t} \widehat{R} \quad \in \mathcal{H}$$

Closed-form expression:

$$F^t = (f^t(X_1), \ldots, f^t(X_n)) \in \mathbb{R}^n$$

$$\begin{cases} F^t &= \left[I_n - \Pi_{s=1}^t (I_n - \alpha K_n) \right] Y, \qquad t \geq 1 \\ F^0 &= 0 \end{cases}$$

with

- $K_n = K/n$: normalized Gram matrix
- $\hat{\mu}_1 \geq \cdots \geq \hat{\mu}_n \geq 0$: nonincreasing eigenvalues of K_n
- α such that $\alpha \hat{\mu}_1 < 1 \to$ Why?

Remark:

GD is a particular instance of the family of spectral filter learning strategies (KRR, spectral cut-off,...)

Proof of the previous result

Hints:

- Calculate the gradient

Proof of the previous result

Hints:

- Calculate the gradient
- From functions to vectors ...

Proof of the previous result

The Big Picture about kernel machines

Focus on the regression problem

Reproducing Kernel Hilbert Space

Iterative learning strategies

Gradient descent algorithm

Stochastic Gradient Descent algorithm

Hints:

- Calculate the gradient
- From functions to vectors ...
- Prove that:

$$F^{t+1} - \mathrm{Y} = (I - \alpha K_n)\left(F^t - \mathrm{Y}\right)$$

Proof of the previous result

Hints:

- Calculate the gradient
- From functions to vectors ...
- Prove that:

$$F^{t+1} - \mathrm{Y} = (I - \alpha K_n) \left(F^t - \mathrm{Y} \right)$$

- Deduce that:

$$\left(F^t - \mathrm{Y} \right) = (I - \alpha K_n)^t \left(F^0 - \mathrm{Y} \right)$$

Limitations of the GD algorithm

$$\begin{cases} F^t & = \left[I_n - \Pi_{s=1}^t (I_n - \alpha K_n) \right] Y, \qquad t \geq 1 \\ F^0 & = 0 \end{cases}$$

Computational aspects

- All the n observations are involved at each step of GD
- With large datasets, becomes no longer tractable
- Requires the use of a fast-to-compute substitute to GD

Limitations of the GD algorithm

$$\begin{cases} F^t &= \left[I_n - \Pi_{s=1}^t \left(I_n - \alpha K_n \right) \right] Y, \qquad t \geq 1 \\ F^0 &= 0 \end{cases}$$

Computational aspects

- All the n observations are involved at each step of GD
- With large datasets, becomes no longer tractable
- Requires the use of a fast-to-compute substitute to GD

$$\nabla_{f^t} \widehat{R} = \sum_{i=1}^{n} \left(\nabla_{f^t} \widehat{R} \right)_i = -\frac{2}{n} \sum_{i=1}^{n} k_{X_i} \left(Y_i - f^t(X_i) \right)$$

Limitations of the GD algorithm

$$\begin{cases} F^t & = \left[I_n - \Pi_{s=1}^t \left(I_n - \alpha K_n \right) \right] Y, \qquad t \geq 1 \\ F^0 & = 0 \end{cases}$$

Computational aspects

- All the n observations are involved at each step of GD
- With large datasets, becomes no longer tractable
- Requires the use of a fast-to-compute substitute to GD

$$\nabla_{f^t} \widehat{R} = \sum_{i=1}^n \left(\nabla_{f^t} \widehat{R} \right)_i = -\frac{2}{n} \sum_{i=1}^n k_{X_i} \left(Y_i - f^t(X_i) \right)$$

Remark:

$\rightarrow$ Stochastic Gradient Descent (SGD) overcomes this limitation

The Stochastic Gradient Descent algorithm

SGD derivation (1/2)

Prediction Error (PE)

$$PE(f) = \mathbb{E}_{(X,Y)}\left[(Y - f(X))^2 \right] = \mathbb{E}_{(X,Y)}\left[(Y - \langle f, k_X \rangle_{\mathcal{H}})^2 \right]$$

Intuition At each step of the iterative algorithm,

$$PE(f) \approx PE(f^t) + \left\langle \nabla_{f^t} PE, f - f^t \right\rangle_{\mathcal{H}}$$

SGD derivation (1/2)

The Big Picture about kernel machines

Focus on the regression problem

Reproducing Kernel Hilbert Space

Iterative learning strategies

Gradient descent algorithm

Stochastic Gradient Descent algorithm

Prediction Error (PE)

$$
\begin{aligned}
PE(f) &= \mathbb{E}_{(X,Y)}\left[(Y - f(X))^2 \right] \\
&= \mathbb{E}_{(X,Y)}\left[(Y - \langle f, k_X \rangle_{\mathcal{H}})^2 \right]
\end{aligned}
$$

Intuition At each step of the iterative algorithm,

$$
PE(f) \approx PE(f^t) + \left\langle \nabla_{f^t} PE, f - f^t \right\rangle_{\mathcal{H}}
$$

Computing the gradient

$$
\nabla_{f^t} PE = \mathbb{E}_{(X,Y)}\left[-2\,(Y - f(X))\, k_X \right]
$$

SGD derivation (1/2)

Prediction Error (PE)

$$\begin{aligned} PE(f) &= \mathbb{E}_{(X,Y)}\left[(Y - f(X))^2 \right] \\ &= \mathbb{E}_{(X,Y)}\left[(Y - \langle f, k_X \rangle_{\mathcal{H}})^2 \right] \end{aligned}$$

Intuition At each step of the iterative algorithm,

$$PE(f) \approx PE(f^t) + \left\langle \nabla_{f^t} PE, f - f^t \right\rangle_{\mathcal{H}}$$

Computing the gradient

$$\nabla_{f^t} PE = \mathbb{E}_{(X,Y)}\left[-2\,(Y - f(X))\, k_X \right]$$

Approximating the gradient

$$\nabla_{f^t} PE \approx -2\,(Y_{i_t} - f(X_{i_t}))\, k_{X_{i_t}} = \left(\nabla_{f^t} \widehat{R} \right)_{i_t}$$

with i_t: Index chosen at random independently of the data

SGD derivation (2/2)

From GD to SGD

$$\text{GD} \qquad \longrightarrow \qquad f^{t+1} = f^t - \alpha \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - f^t(X_i)\right)\cdot k_{X_i}}_{=1/2\nabla_{f^t}\widehat{R}}$$

SGD derivation (2/2)

From GD to SGD

$$\text{GD} \quad \longrightarrow \quad f^{t+1} = f^t - \alpha \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - f^t(X_i)\right) \cdot k_{X_i}}_{=1/2\nabla_{f^t}\widehat{R}}$$

$$\text{SGD} \quad \longrightarrow \quad f^{t+1} = f^t - \alpha\left(Y_{i_t} - f^t(X_{i_t})\right) \cdot k_{X_{i_t}}$$

with $(i_t)_{t\in\mathbb{N}_+}$ sequence of random indices

SGD derivation (2/2)

From GD to SGD

$$\text{GD} \quad \longrightarrow \quad f^{t+1} = f^t - \alpha \underbrace{\frac{1}{n}\sum_{i=1}^{n} \left(Y_i - f^t(X_i)\right) \cdot k_{X_i}}_{=1/2 \nabla_{f^t} \widehat{R}}$$

$$\text{SGD} \quad \longrightarrow \quad f^{t+1} = f^t - \alpha \left(Y_{i_t} - f^t(X_{i_t})\right) \cdot k_{X_{i_t}}$$

with $(i_t)_{t \in \mathbb{N}_+}$ sequence of random indices

Theorem

With i_t chosen uniformly at random,

$$\mathbb{E}_{i_t}\left[\left(\nabla_{f^t} \widehat{R}\right)_{i_t}\right] = \frac{1}{n}\sum_{i=1}^{n} \left(Y_i - f^t(X_i)\right) \cdot k_{X_i} = \nabla_{f^t} \widehat{R}$$

SGD: unbiased estimator of GD at each step

Homeworks

- Implement both GD and SGD algorithms
- Illustrate the behavior on a regression problem with $d = 2$ and $\mathcal{X} = [0, 1]^2$ (convex problem):
  - Nb of iterations until convergence
  - Total computation time until convergence
  - Influence of step size
  - Influence of initialization value
- Provide graphs for illustrating each aspect