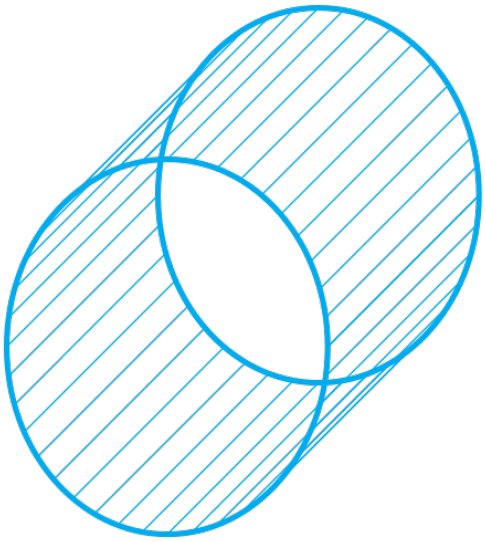
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN ĐẠI HỌC
QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

KHOA TOÁN TIN HỌC

o0o

BÁO CÁO XỬ LÝ ĐA CHIỀU



Khoa Toán - Tin học

Fac. of Math. & Computer Science

KERNEL METHODS

Giảng viên hướng dẫn: **Kha Tuấn Minh**

Nhóm thực hiện: **Nhóm 7**

Mục lục

1 Thành viên nhóm

1. Phùng Thị Điệp - 19110281
2. Hồ Diệp Thanh Thảo - 19110183
3. Vũ Đức Huy - 19110088

2 Nội dung

2.1 Kernels, RKHS

2.1.1 Introduction

Kernel methods là một lớp các thuật toán được sử dụng để phân tích mẫu, phân loại trong máy học, phương pháp phổ biến nhất của lớp thuật toán này là **Support Vector Machines (SVMs)**. **Kernel methods** liên quan đến việc sử dụng các phân loại tuyến tính **(linear classifiers)** để giải quyết các vấn đề phi tuyến **(nonlinear problems)** bằng cách sử dụng một hàm toán học gọi là **kernel function** để chuyển dữ liệu đầu vào sang một không gian khác có số chiều cao hơn.

Kernel methods là một công cụ mạnh vì nó có thể tìm hiểu các mối quan hệ phức tạp giữa các features. Ngoài ra, nó có thể xử lý dữ liệu nhiều chiều một cách hiệu quả mà thường yêu cầu ít tài nguyên tính toán hơn so với các kỹ thuật máy học khác. Một số ví dụ của **kernel methods** như **SVMs, Gaussian Processes, kernel PCA,...**

2.1.2 Positive Definite Kernels

Kernel methods yêu cầu một hàm tương đồng **(similarity function)** trên tất cả các cặp điểm dữ liệu:

$$k : \chi \times \chi \to \mathbb{R}, \qquad (x, x') \mapsto k(x, x')$$

thỏa ∀ x, x' ∈ χ,
Hàm k được gọi là hàm tương đồng **similarity function** hay **kernel**.
Ta biểu diễn sự tương đồng giữa các điểm dữ liệu dưới dạng ma trận có kích thước $n \times n$ như sau:

$$[K]_{ij} := k(x_i, x_j)$$

K được gọi là ma trận tương đồng **similarity matrix**.

. Nhận xét: Với việc sử dụng ma trận K, thuật toán hoạt động được với bất kỳ loại dữ liệu (vectors, chuỗi,...).

Với kích thước tập dữ liệu lớn, khả năng mở rộng của thuật toán kém **(poor scalability)** vì tính toán và lưu trữ ma trận K đòi hỏi độ phức tạp là $O(n^2)$.

Định nghĩa. Positive Definite Kernels
Một kernel xác định dương trên một tập χ là một hàm đối xứng **(symmetric function)** $k : \chi \times \chi \to \mathbb{R}$

$$\forall (x, x') \in \chi^2, \;\; k(x, x') = k(x', x)$$

và thỏa $\forall N \in \mathbb{N}, (x_1, x_2, ..., x_N) \in \chi^N$ và $(a_1, a_2, ..., a_N) \in \mathbb{R}^N$:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j k(x_i, x_j) \geq 0$$

. Nhận xét: kernel k xác định dương khi và chỉ khi $\forall N \in \mathbb{N} v (x_1, x_2, ..., x_N) \in \chi^N$, ma trận tương đồng K là ma trận nửa xác định dương **(positive semidefinite)**.

Đầu vào của kernel methods là ma trận K được định nghĩa như trên.

. Bổ đề: kernel xác định dương với số thực
Cho $\chi = \mathbb{R}$. Hàm $k : \mathbb{R}^2 \mapsto \mathbb{R}$ được xác định như sau:

$$\forall (x, x') \in \mathbb{R}^2, k(x, x') = xx'$$

là xác định dương.

* Chứng minh:
$xx' = x'x$
$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j x_i x_j = \left(\sum_{i=1}^{N} a_i x_i \right)^2 \geq 0$

. Bổ đề: kernel xác định dương với vectors

Cho $\chi = \mathbb{R}^d$. Hàm $k : \chi^2 \mapsto \mathbb{R}$ được xác định như sau:

$$\forall (x, x') \in \chi^2, k(x, x') = \langle x, x' \rangle_{\mathbb{R}^d}$$

là xác định dương (Hàm này thường được gọi là **linear kernel**.

* Chứng minh:
$\overline{\langle x, x' \rangle_{\mathbb{R}^d}} = \langle x', x \rangle_{\mathbb{R}^d}$
$\sum_{i=1}^N \sum_{j=1}^N a_i a_j \langle x_i, x_j \rangle_{\mathbb{R}^d} = \| \sum_{i=1}^N a_i x_i \|^2_{\mathbb{R}^d} \geq 0$

. Bổ đề: A more ambitious p.d. kernel

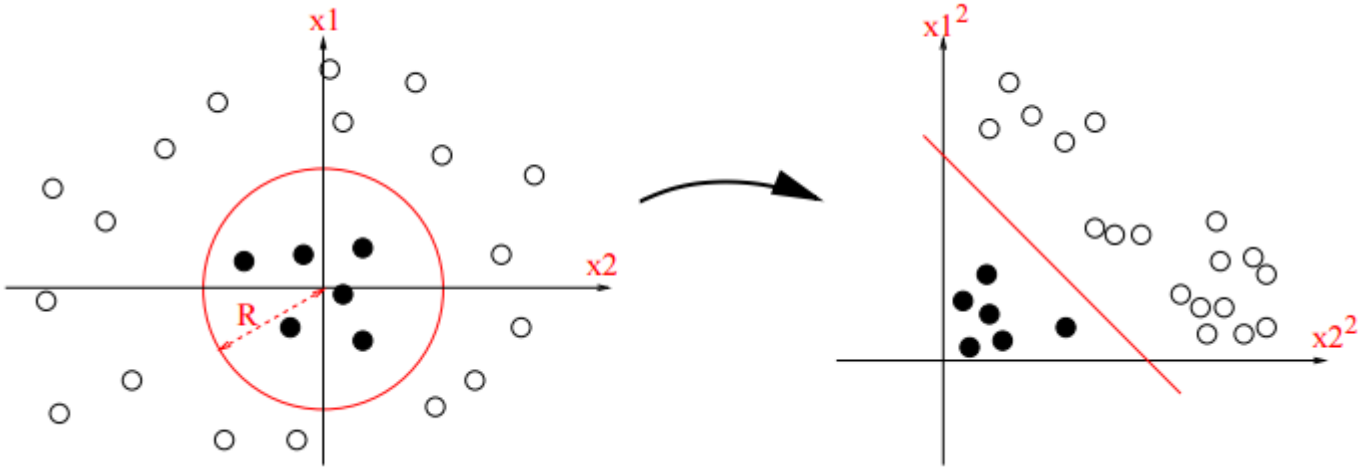
Cho χ là một tập bất kỳ và $\phi : \chi \mapsto \mathbb{R}^d$. Khi đó, hàm $k : \chi^2 \mapsto \mathbb{R}$ được xác định như sau:

$$\forall (x, x') \in \chi^2, k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathbb{R}^d}$$

là xác định dương.

* Chứng minh:
$\overline{\langle \phi(x), \phi(x') \rangle_{\mathbb{R}^d}} = \langle \phi(x'), \phi(x) \rangle_{\mathbb{R}^d}$
$\sum_{i=1}^N \sum_{j=1}^N a_i a_j \langle \phi(x_i), \phi(x_j) \rangle_{\mathbb{R}^d} = \| \sum_{i=1}^N a_i \phi(x_i) \|^2_{\mathbb{R}^d} \geq 0$

Ví dụ: Polynomial kernel



Với $x = (x_1, x_2)^T \in \mathbb{R}^2$, đặt $\phi(x) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2) \in \mathbb{R}^3$

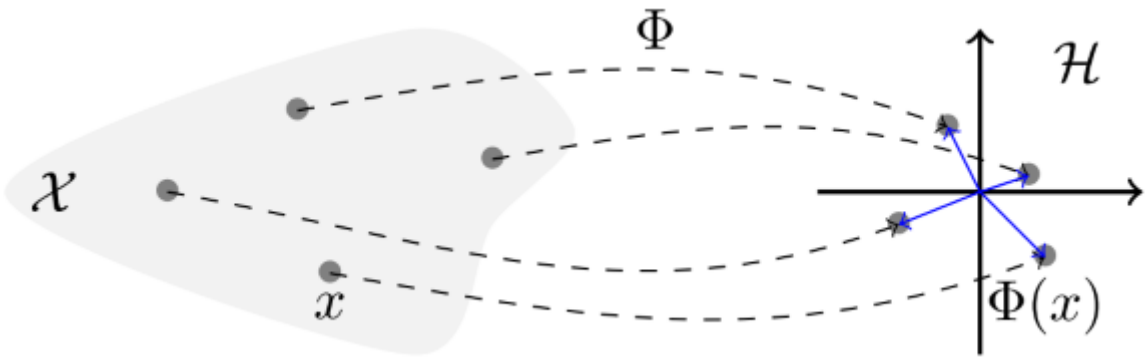
$$\begin{aligned} k(x, x') &= x_1^2 x_1'^2 + 2 x_1 x_2 x_1' x_2' + x_2^2 x_2'^2 \\ &= (x_1 x_1' + x_2 x_2')^2 \\ &= \langle x, x' \rangle^2_{\mathbb{R}^2} \end{aligned}$$

Định lý (Aronszajn, 1950)**.** k là kernel xác định dương trên tập χ khi và chỉ khi tồn tại một không gian Hilbert $\mathcal{H}$ và một ánh xạ:

$$\phi : \chi \mapsto \mathcal{H}$$

sao cho $\forall x, x' \in \chi$:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$$



2.1.3 Reproducing Kernel Hilbert Spaces (RKHS)

Định nghĩa. Cho χ là một tập hợp và $\mathcal{H} \subset \mathbb{R}^{\chi}$ là một lớp hàm tạo thành không gian Hilbert với tích trong **inner product** $\langle ., . \rangle_{\mathcal{H}}$. Hàm $k : \chi^2 \mapsto \mathbb{R}$ được gọi là **reproducing kernel (r.k.)** of $\mathcal{H}$ nếu:

1. $\mathcal{H}$ chứa tất cả hàm có dạng
$$\forall x \in \chi, k_x : t \mapsto k(x, t)$$
2. Với mỗi $x \in \chi$ và $f \in \mathcal{H}$ thì **reproducing property**:
$$f(x) = \langle f, k_x \rangle_{\mathcal{H}}$$

.

Nếu reproducing kernel tồn tại thì $\mathcal{H}$ được gọi là **reproducing kernel Hilbert space (RKHS)**

Định lý. Không gian Hilbert $\mathcal{H} \subset \mathbb{R}^{\chi}$ là một RKHS khi và chỉ khi với bất kỳ $x \in \chi$, ánh xạ tuyến tính:

$$\begin{aligned} F :& \mathcal{H} \to \mathbb{R} \\ & f \mapsto f(x) \end{aligned}$$

là liên tục.

* Chứng minh: **Chiều thuận**

Nếu tồn tại một reproducing kernel k thì với bất kỳ $(x, f) \in \chi \times \mathcal{H}$:

$$\begin{aligned} |f(x)| &= |\langle f, k_x \rangle_{\mathcal{H}}| \\ &\leq \| f \|_{\mathcal{H}} . \| k_x \|_{\mathcal{H}} \qquad (Cauchy - Schwarz) \\ &\leq \| f \|_{\mathcal{H}} . k(x, x)^{\frac{1}{2}} \end{aligned}$$

do $\| k_x \|^2_{\mathcal{H}} = \langle k_x, k_x \rangle_{\mathcal{H}} = k(x, x)$. Vì vậy, $f \in \mathcal{H} \mapsto f(x) \in \mathbb{R}$ là một ánh xạ tuyến tính liên tục.

Chiều ngược

Giả sử với $x \in \chi$ bất kỳ, ánh xạ tuyến tính $f \in \mathcal{H} \mapsto f(x)$ liên tục thì theo **Riesz representation theorem**, tồn tại duy nhất $g_x \in \mathcal{H}$ sao cho:

$$f(x) = \langle f, g_x \rangle_{\mathcal{H}}$$

. Khi đó hàm $k(x, y) = g_x(y)$ là một reproducing kernel của $\mathcal{H}$.

Định lý. Một hàm k: $\chi \times \chi \to \mathbb{R}$ là xác định dương khi và chỉ khi nó là một reproducing kernel.

* Chứng minh: **Chiều thuận: Một reproducing kernel là xác định dương**

- Một reproducing kernel là đối xứng bởi vì với $(x, y) \in \chi^2$ bất kỳ:
$$k(x, y) = \langle k_x, k_y \rangle_{\mathcal{H}} = \langle k_y, k_x \rangle_{\mathcal{H}} = k(y, x)$$

- k xác định dương vì với bất kỳ $N \in \mathbb{N}, (x_1, x_2, ..., x_N) \in \chi^N$ và $(a_1, a_2, ..., a_N) \in \mathbb{R}^N$:

$$\begin{aligned}\sum_{i,j=1}^{N} a_i a_j k(x_i, x_j) &= \sum_{i,j=1}^{N} a_i a_j \langle k_{x_i}, k_{x_j} \rangle_{\mathcal{H}} \\ &= \| \sum_{i=1}^{N} a_i k_{x_i} \|^2_{\mathcal{H}} \\ &\geq 0\end{aligned}$$

  * Chứng minh: **Chiều ngược: Một kernel xác định dương là một reproducing kernel**

- Đặt $\mathcal{H}_0$ là một không gian con vector của $\mathbb{R}^{\chi}$ được mở rộng bởi hàm $\{k_x\}_{x \in \chi}$
- Với $f, g \in \mathcal{H}_0$ cho bởi:

$$f = \sum_{i=1}^{m} a_i k_{x_i}, \qquad g = \sum_{j=1}^{n} b_j k_{y_j}$$

  Đặt:

$$\langle f, g \rangle_{\mathcal{H}0} := \sum_{i,j} a_i b_j k(x_i, y_j)$$

- $\langle f, g \rangle_{\mathcal{H}0}$ không phụ thuộc vào khai triển của f và g vì:

$$\langle f, g \rangle_{\mathcal{H}0} = \sum_{i=1}^{m} a_i g(x_i) = \sum_{j=1}^{n} b_j f(y_j)$$

- Điều này cũng chỉ ra rằng $\langle ., . \rangle_{\mathcal{H}0}$ là một dạng song tuyến tính đối xứng.
- Điều này cũng chỉ ra rằng với bất kỳ $x \in \chi$ và $f \in \mathcal{H}_0$:

$$\langle f, k_x \rangle_{\mathcal{H}0} = f(x)$$

- Giả sử k xác định dương:

$$\| f \|^2_{\mathcal{H}_0} = \sum_{i,j=1}^{m} a_i a_j k(x_i, x_j) \geq 0$$

- Theo Cauchy-Schwarz, suy ra $\forall x \in \chi$:

$$|f(x)| = |\langle f, k_x \rangle_{\mathcal{H}0}| \leq \| f \|_{\mathcal{H}\ 0} . k(x, x)^{\frac{1}{2}}$$

  Vì vậy, $\| f \|_{\mathcal{H}\ 0} = 0 \Longrightarrow f = 0$
- $\mathcal{H}_0$ là một không gian tiền Hilbert được trang bị tích trong $\langle ., . \rangle_{\mathcal{H}0}$
- Với dãy Cauchy bất kỳ $(f_n)_{n \geq 0}$ trong $(\mathcal{H}_0, \langle ., . \rangle_{\mathcal{H}0}$, ta lưu ý:

$$\forall (x, m, n) \in \chi \times \mathbb{N}^2, \quad |f_m(x) - f_n(x) \leq \| f_m - f_n \|_{\mathcal{H}\ 0} . k(x, x)^{\frac{1}{2}}|$$

  là dãy hội tụ.
- Nếu thêm vào $\mathcal{H}_0$ các hàm được định nghĩa là dãy Cauchy hội tụ điểm thì không gian trở nên hoàn chỉnh và vì vậy là một không gian Hilbert với k là reproducing kernel.

2.2 Smoothness functional, Kernel trick, Representer theorem

2.2.1 Smoothness functional (Hàm mịn)

Theo Cauchy-Schwarz, với mọi hàm số $f \in \mathcal{H}$ và hai điểm bất kỳ $x, x' \in X$:

$$|f(x) - f(x')| = |\langle f, K(x) - K(x')\rangle_{\mathcal{H}}| \leq \|f\|_{\mathcal{H}} \cdot \|K(x) - K(x')\|_{\mathcal{H}} = \|f\|_{\mathcal{H}} \cdot d_K(x, x')$$

Trong đó $|f(x) - f(x')|$: Giá trị tuyệt đối giữa các giá trị hàm $f(x)$ và $f(x')$.
$|\langle f, K(x) - K(x')\rangle_{\mathcal{H}}|$: Giá trị tuyệt đối của tích bên trong giữa hàm f và giá trị chênh lệch của hạt nhân $K(x) - K(x')$.
$\|f\|_{\mathcal{H}}$: Chuẩn (hoặc độ lớn) của hàm f trong RKHS.
$\|K(x) - K(x')\|_{\mathcal{H}}$: Chuẩn của giá trị chênh lệch giữa các hạt nhân $K(x)$ và $K(x')$ trong RKHS.
$d_K(x, x')$: Khoảng cách giữa các điểm x và x' do hàm kernel K tạo ra.
Bất đẳng thức trên nói rằng giá trị tuyệt đối của chênh lệch giữa $f(x)$ và $f(x')$ bị giới hạn bởi tích của chuẩn của f trong RKHS và khoảng cách giữa các hạt nhân tại x và x'. Nói cách khác, giá trị hàm f thay đổi giữa x và x' dựa vào chuẩn của f và khoảng cách do hàm kernel tạo ra.
Chuẩn của hàm trong RKHS dựa vào tốc độ thay đổi của hàm trên X đối với dạng hình học được xác định bởi nhân (Lipschitz với hằng số $\|f\|_{\mathcal{H}}$).
Lưu ý: Chuẩn nhỏ $\Rightarrow$ Biến đổi chậm
Tóm tắt về Kernel và RKHS: Kernel dương (p.d.) có thể coi như tích vô hướng sau khi nhúng không gian dữ liệu X vào một không gian Hilbert. Vì vậy, một kernel dương p.d. xác định một phép đo trên X. Một cách hiện thực của việc nhúng này là không gian tái sinh theo kernel (RKHS), có hiệu lực mà không có ràng buộc về không gian X hay về kernel. RKHS là một không gian các hàm trên X. Chuẩn của một hàm trong RKHS liên quan đến mức độ mịn màng của nó đối với phép đo được xác định bởi kernel trên X.
Bây giờ chúng ta sẽ xem một số ứng dụng của kernel và RKHS trong thống kê, trước khi quay trở lại vấn đề lựa chọn kernel.
Ta có 2 phương pháp kernel:

- **Kỹ thuật kernel**: Dựa trên việc biểu diễn các kernel dương p.d. như tích vô hướng.
- **Định lý representer**: Dựa trên một số tính chất của hàm chính quy hóa được xác định bởi chuẩn của RKHS.

2.2.2 The Kernel trick (Kỹ thuật kernel)

> **Mệnh đề**
> Bất kì thuật toán xử lý các vector hữu hạn chiều chỉ dựa trên tích vô hướng của các cặp vector có thể được áp dụng cho các vector có thể có số chiều vô hạn trong không gian đặc trưng của một kernel dương (p.d. kernel) bằng cách thay thế giá trị tích vô hướng bằng giá trị kernel.

Chú ý:

- Kernel chính là tích vô hướng trong không gian đặc trưng
- Kỹ thuật này có ứng dụng thực tế rất lớn.
- Các vector trong không gian đặc trưng chỉ được xử lý một cách ngầm định, thông qua tích vô hướng của cặp vector.
- Khoảng cách trong không gian đặc trưng:
  Cho X là không gian dữ liệu; x_1, x_2 là hai điểm dữ liệu trong không gian X; $\phi(x)$ là ánh xạ phi (feature mapping) của X; $d(x_1, x_2)$: khoảng cách giữa x_1 và x_2 trong không gian ϕ ta có công thức

$$d_K(x_1, x_2)^2 = K(x_1, x_1) + K(x_2, x_2) - 2K(x_1, x_2)$$

- Khoảng cách trong kernel Gaussian: Khoảng cách giữa hai điểm x và y trong không gian đặc trưng được cho bởi

$$d_K(x, y) = \sqrt{2\left(1 - e^{-\frac{\|x-y\|^2}{2\sigma^2}}\right)}$$

- Khoảng cách giữa một điểm và một tập hợp
  Cho $S = (x_1, \ldots, x_n)$ là một tập hợp hữu hạn các điểm trong X. Khoảng cách giữa một điểm x bất kỳ trong X và tập hợp S được xác định bởi

$$d_K(x, S) = \sqrt{K(x,x) - \frac{2}{n}\sum_{i=1}^{n} K(x, x_i) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} K(x_i, x_j)}$$

- Tâm dữ liệu trong không gian đặc trưng
  Cho $S = (x_1, \ldots, x_n)$ là một tập hợp hữu hạn các điểm trong X cho biết một kernel dương K với K là ma trận Gram n x n: $[K]_{ij} = K(x_i, x_j)$ và $\mu = \frac{1}{n}\sum_{i=1}^{n} \Phi(x_i)$ là trọng tâm của nó, đặt $u_i = \Phi(x_i) - \mu$ $(i = 1, \ldots, n)$. Ta tính được tâm ma trận Gram mới bởi công thức

$$K^c = K - UK - KU + UKU = (I - U)K(I - U), U_{ij} = \frac{1}{n}, 1 \leq i, j \leq n$$

Tóm lại Kernel trick có ứng dụng quan trọng. Nó có thể được sử dụng để thu được phiên bản phi tuyến của các thuật toán tuyến tính nổi tiếng, ví dụ như thay thế tích vô hướng bằng một kernel Gaussian. Nó còn được sử dụng để áp dụng các thuật toán cổ điển cho dữ liệu không gian vector (ví dụ như chuỗi, đồ thị) bằng cách thay thế tích vô hướng bằng một kernel hợp lệ cho dữ liệu. Kernel trick cho phép trong một số trường hợp nhúng không gian ban đầu vào một không gian đặc trưng lớn hơn và liên quan đến các điểm trong không gian đặc trưng mà không có hình ảnh trước (ví dụ như trọng tâm).

2.2.3 Representer theorem (Định lý representer)

Định lý Representer
Cho X là một tập hợp có kernel K xác định dương, H là không gian Hilbert tương ứng (RKHS), và $S = \{x_1, \ldots, x_n\} \subseteq X$ là một tập hợp hữu hạn các điểm trong X.
Giả sử $\Psi : \mathbb{R}^{n+1} \to \mathbb{R}$ là một hàm của $n + 1$ biến, tăng theo cấp số nhân với biến cuối cùng.
Khi đó:

$$\min_{f \in H} \Psi(f(x_1), \ldots, f(x_n), \|f\|_H)$$

được biểu diễn dưới dạng:

$$\forall x \in X, f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) = \sum_{i=1}^{n} \alpha_i K_{x_i}(x)$$

Trong đó, α_i là các hệ số và $K(x_i, x)$ là kernel tại cặp điểm (x_i, x).
Nói cách khác, f nằm trong một không gian con có số chiều hữu hạn:

$$f \in \text{Span}(K(x_1), \ldots, K(x_n))$$

2.3 Kernel ridge and logistic regression

2.3.1 Kernel ridge regression (KRR)

- Xét H là một RKHS liên quan đến kernel xác định dương K trên X.
- KRR (Kernel Ridge Regression) được thu được bằng cách điều chỉnh MSE bằng chuẩn RKHS:

$$\hat{f} = \arg\min_{f \in H} \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda\|f\|_H^2 \tag{2}$$

Trong đó, ảnh hưởng của tham số λ là ngăn chặn việc overfitting bằng cách xét các hàm không liên tục.

- Theo định lý representer, từ (2) có thể viết đơn giản như sau:

$$\hat{f}(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

Giải bài toán KRR:
Cho $y = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$
Cho $\alpha = (\alpha_1, \ldots, \alpha_n)^T \in \mathbb{R}^n$
Cho K là ma trận Gram kích thước $n \times n$: $K_{ij} = K(x_i, x_j)$
Ta có thể viết lại như sau:

$$(\hat{f}(x_1), \ldots, \hat{f}(x_n))^T = K\alpha$$

Tương đương với:

$$\|\hat{f}\|_{\mathcal{H}}^2 = \alpha^T K \alpha$$

KRR (2) tương đương với:

$$\arg\min_{\alpha \in \mathbb{R}^n} \frac{1}{n}(K\alpha - y)^T(K\alpha - y) + \lambda \alpha^T K \alpha$$

Đây là một hàm lồi và khả vi của α . Vì vậy, giá trị nhỏ nhất có thể được tìm bằng cách đặt đạo hàm theo α bằng không:

$$0 = \frac{2}{n}K(K\alpha - y) + 2\lambda K\alpha = K[(K + \lambda nI)\alpha - y]$$

Với $\lambda > 0$, $K + \lambda nI$ là khả nghịch (vì K là nửa xác định dương), vì vậy:

$$\alpha = (K + \lambda nI)^{-1} y$$

Chú ý: Liên kết với "standard"ridge regression

Lemma về đảo ma trận
Đối với bất kỳ ma trận B và C, và $\gamma > 0$, ta có:

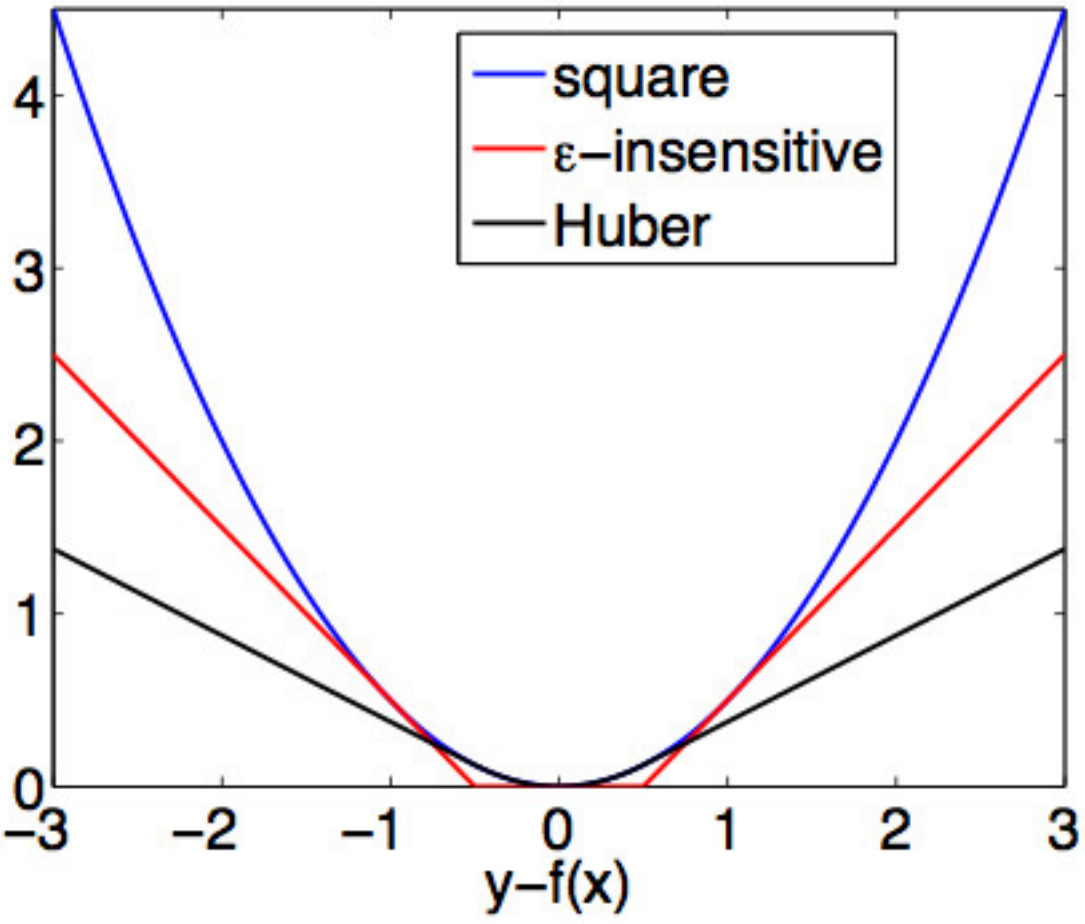
$$B(CB + \gamma I)^{-1} = (BC + \gamma I)^{-1} B$$

Từ đó, ta suy ra:

$$w_{\text{RR}} = (X^T X + \lambda nI)^{-1} X^T y = X^T (XX^T + \lambda nI)^{-1} y = w_{\text{KRR}}$$

Việc đảo ma trận gây tốn kém nên ta thực hiện: KRR khi $d > n$ (số chiều cao) và RR khi $d < n$ (nhiều điểm).
* Hồi quy mạnh mẽ (Robust regression):

- Bình phương sai số $l(t; y) = (t - y)^2$ nhạy cảm với các giá trị ngoại lai.
- Nhiều hàm lồi khác tồn tại hồi quy, ví dụ:



- Bất kỳ hàm lồi nào cũng dẫn đến một phương pháp kernel hợp lệ, thường được giải quyết bằng tối ưu số học vì thường không có giải pháp phân tích ngoài bình phương hàm lồi.

* Hồi quy có trọng số (Weighted regression):

Cho các trọng số $W_1, \ldots, W_n$ thuộc $\mathbb{R}$, một biến thể của hồi quy ridge là áp dụng trọng số khác nhau cho các sai số ở các điểm khác nhau:

$$\arg\min_{f \in H} \frac{1}{n} \sum_{i=1}^{n} W_i (y_i - f(x_i))^2 + \lambda \|f\|_H^2$$

Theo định lý representer, giải pháp được biểu diễn bởi $f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$, trong đó α giải bài toán tối ưu sau, với $W = \text{diag}(W_1, \ldots, W_n)$:

$$\arg\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} (K\alpha - y)^\top W (K\alpha - y) + \lambda \alpha^\top K \alpha$$

Hình thức này cho phép áp dụng các trọng số khác nhau cho mỗi điểm dữ liệu, mang lại tính linh hoạt trong việc xử lý mức độ quan trọng hoặc không chắc chắn khác nhau liên quan đến các quan sát.
Sau khi đặt gradient bằng không, ta có:

$$0 = \frac{2}{n} K W^{\frac{1}{2}} \left[\left(W^{\frac{1}{2}} K W^{\frac{1}{2}} + n\lambda I \right) W^{-\frac{1}{2}} \alpha - W^{\frac{1}{2}} y \right]$$

Vì vậy,

$$\left(W^{\frac{1}{2}} K W^{\frac{1}{2}} + n\lambda I \right) W^{-\frac{1}{2}} \alpha - W^{\frac{1}{2}} y = 0$$

Do đó

$$\alpha = W^{\frac{1}{2}} \left(W^{\frac{1}{2}} K W^{\frac{1}{2}} + n\lambda I \right)^{-1} W^{\frac{1}{2}} Y$$

2.3.2 Kernel logistic regression (KLR)

Ta có logistic loss:
$\text{logistic}(f(x), y) = -\ln p(y \mid f(x)) = \ln\left(1 + e^{-yf(x)}\right)$

$$\hat{f} = \arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \text{logistic}\left(f(x_i), y_i\right) + \frac{\lambda}{2}\|f\|_H^2$$

$$= \arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \ln\left(1 + \exp\left(-y_i f(x_i)\right)\right) + \frac{\lambda}{2}\|f\|_H^2$$

Có thể hiểu như một ước lượng hợp lý cực đại có điều kiện được điều chỉnh.
Giải phương trình KLR:
Theo định lý representer:

$$\hat{f}(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

và ta có:

$$\left(\hat{f}(x_1), \ldots, \hat{f}(x_n)\right)^T = K\alpha \quad \text{và} \quad \|\hat{f}\|_H^2 = \alpha^T K\alpha$$

Để tìm giá trị α, chúng ta cần giải quyết bài toán tối ưu sau đây:

$$\min_{\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} \ln\left(1 + e^{-y_i[K\alpha]_i}\right) + \frac{\lambda}{2}\alpha^\top K\alpha$$

Đây là một bài toán tối ưu hóa mượt và lồi, có thể được giải bằng nhiều phương pháp. Sử dụng phương pháp Newton có xấp xỉ J bằng một hàm bậc hai và giải bài toán bậc hai. Ta được:

$$\nabla^2 J(\alpha) = \frac{1}{n} KW(\alpha)K + \lambda K$$

trong đó $W(\alpha) = \text{diag}(W_1(\alpha), \ldots, W_n(\alpha))$. Ta có:

$$J_q(\alpha) = J(\alpha_0) + (\alpha - \alpha_0)^T \nabla J(\alpha_0) + \frac{1}{2}(\alpha - \alpha_0)^T \nabla^2 J(\alpha_0)(\alpha - \alpha_0)$$

Với $P = P(\alpha_0)$ và $W = W(\alpha_0)$:

- $\alpha^T \nabla J(\alpha_0) = \frac{1}{n}\alpha^T KPy + \lambda\alpha^T K\alpha_0$
- $\frac{1}{2}\alpha^T \nabla^2 J(\alpha_0)\alpha = \frac{1}{2n}\alpha^T KWK\alpha + \frac{\lambda}{2}\alpha^T K\alpha$
- $-\alpha^T \nabla^2 J(\alpha_0)\alpha_0 = -\frac{1}{n}\alpha^T KWK\alpha_0 - \lambda\alpha^T K\alpha_0$

Suy ra được:

$$2J_q(\alpha) = \frac{1}{n}(K\alpha - z)^T W(K\alpha - z) + \lambda\alpha^T K\alpha + C$$

Đây là một bài toán chuẩn Ridge Regression có trọng số (WKRR)!
Tóm lại, để giải bài toán KLR là lặp lại việc giải các bài toán WKRR cho đến khi đạt được sự hội tụ:

$$\alpha^{t+1} \leftarrow solveWKRR(K, W^t, z^t)$$

Quy tắc cập nhật cho W^t và z^t từ α^t có thể được biểu diễn như sau:

- $m_i \leftarrow \left[K\alpha^T\right]_i$
- $P_i^t \leftarrow l'_{logistic(y_i m_i)} = -\alpha(-y_i m_i)$
- $W_i^t \leftarrow l''_{logistic(y_i m_i)} = \sigma(m_i)\sigma(-m_i)$
- $z_i^t \leftarrow m_i - \frac{P_i^t y_i}{W_i^t} = m_i + \frac{y_i}{\sigma(y_i m_i)}$

Đây là phương pháp nổi tiếng dùng kernel của iteratively reweighted least-square (IRLS) để giải quyết bài toán hồi quy tuyến tính logistic.

2.4 Unsupervised learning, kernel PCA, K-means, CCA

2.4.1 PCA

PCA là một thuật toán cổ điển trong thống kê đa biến để xác định các thành phần chính nhằm thu thập phương sai tối đa từ một tập hợp các vector $S = x_1, \ldots, x_n$ ($x_i \in \mathbb{R}^d$).
Có nhiều ứng dụng trong việc giảm chiều dữ liệu, tìm kiếm cấu trúc tương quan và trực quan hóa dữ liệu. Nó cung cấp một cách đơn giản và ngắn gọn để mô tả cấu trúc dữ liệu ban đầu thông qua các thành phần chính.

Tuy nhiên, PCA chỉ hoạt động hiệu quả trên dữ liệu tuyến tính và thường hoạt động trên dữ liệu lấy số 0 làm trung tâm.

$$\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = 0$$

Trong đó x là một trong số N đa biến hoạt động bằng cách chéo hóa ma trận. Phép chiếu vuông góc xuống hướng $\mathbf{w} \in \mathbb{R}^d$ là một hàm được định nghĩa trên không gian vector dòng $\mathbb{R}^d$, có công thức là:

$$h_{\mathbf{w}}(\mathbf{x}) = \mathbf{x}\frac{\mathbf{w}}{||\mathbf{w}||}$$

với $\mathbf{x}$ là một vector cột bất kỳ trong $\mathbb{R}^d$ và $\mathbf{w}^T$ là phép chuyển vị của $\mathbf{w}$. Phương sai thực nghiệm được thu thập bởi $\mathrm{h}_{\mathbf{w}}(\mathbf{x})$ là:

$$\hat{var}(h_{\mathbf{w}}) := \frac{1}{n}\sum_{n=1}^{n} h_{\mathbf{w}}(\mathbf{x}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{(\mathbf{x}_i\mathbf{w})^2}{||\mathbf{w}||^2}$$

Là một ước lượng thực nghiệm về phương sai $\hat{var}(h_{\mathbf{w}})$ để đo lường mức độ biến động của các giá trị đã được chiếu trực giao lên hướng w trong tập dữ liệu.

Hướng chính thứ i ($\mathbf{w}_i$) được định nghĩa bởi công thức sau:

$$\mathbf{w}_i = \underset{\mathbf{w}(\mathbf{w}_1,\ldots,\mathbf{w}_{i-1})}{argmax} \quad \hat{var}\,(h_{\mathbf{w}}) \quad s.t \quad ||\mathbf{w}|| = 1$$

Xác định hướng chính thứ i ($\mathbf{w}_i$), chúng ta tìm một hướng $\mathbf{w}$ sao cho khi chiếu trực giao dữ liệu lên $\mathbf{w}$, phương sai ước lượng (var ^) của kết quả chiếu là tối đa. Đồng thời, $\mathbf{w}$ cũng phải vuông góc với các hướng chính đã xác định trước đó ($\mathbf{w}_1, \mathbf{w}2, \ldots, \mathbf{w}i - 1$). Hơn nữa, $\mathbf{w}$ được xác định như một vector đơn vị, có độ dài bằng 1.

Cho X là ma trận dữ liệu kích thước $n \times d$, trong đó các hàng là các vector $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Ta có thể viết lại công thức như sau:

$$\hat{\mathrm{var}}(h_{\mathbf{w}}) = \frac{1}{n}\sum_{i=1}^{n}\frac{(\mathbf{x}_i^{\mathsf{T}}\mathbf{w})^2}{\|\mathbf{w}\|^2} = \frac{1}{n}\frac{\mathbf{w}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w}}{\mathbf{w}^{\mathsf{T}}\mathbf{w}}$$

Các vector riêng liên tiếp của ma trận $\mathbf{X}^T\mathbf{X}$ được xếp hạng theo thứ tự giảm dần của các giá trị riêng. Qua việc tối ưu hàm mục tiêu $\hat{\mathrm{var}}(h_{\mathbf{w}})$, chúng ta có thể tìm ra vectơ $\mathbf{w}$ tương ứng với giá trị riêng lớn nhất của ma trận hiệp phương sai $\mathbf{X}^{\mathsf{T}}\mathbf{X}$. Các vectơ $\mathbf{w}$ tương ứng với giá trị riêng lớn nhất đại diện cho các thành phần chính quan trọng nhất trong dữ liệu, và chúng giữ lại phần lớn sự biến thiên của dữ liệu. Chúng ta có biểu thức sau:

$$\mathbf{w}_i = \underset{\mathbf{w}\perp(\mathbf{w}1,\ldots,\mathbf{w}i-1)}{\arg\max} \quad \mathbf{w}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w} \quad \text{s.t.} \quad |\mathbf{w}| = 1$$

Việc giảm chiều dữ liệu trong PCA thường dựa trên việc chọn ra một số thành phần chính đầu tiên, có giá trị riêng lớn nhất, để giải thích dữ liệu một cách tốt nhất trong không gian chiều thấp hơn. Như vậy, chỉ sử dụng một số thành phần chính đầu tiên có thể giảm kích thước của dữ liệu và loại bỏ thông tin ít quan trọng, trong khi vẫn giữ lại các đặc trưng quan trọng của dữ liệu.

2.4.2 Kernel PCA

Với các dữ liệu tuyến tính, PCA cho ra các thành phần chính có hình dạng đơn giản như hình tròn hoặc khối cầu. Tuy nhiên, khi áp dụng PCA trực tiếp lên các dữ liệu phi tuyến tính không tuân theo một hình dạng cố định thì PCA không thể đạt được kết quả tương tự. Trong trường hợp này, kernel PCA sử dụng các hàm

kernel để ánh xạ dữ liệu vào không gian cao chiều hơn, PCA được áp dụng để tính toán các thành phần chính có cấu trúc phân tán không đồng đều trong dữ liệu.

Cho $x_1, \ldots, x_n$ là một tập hợp các điểm dữ liệu trong X; $K : X \times X \to \mathbb{R}$ là một hàm kernel xác định dương và H là không gian không gian Hilbert của kernel tương ứng (RKHS). Giả sử rằng các giá trị trung bình của các điểm dữ liệu đã được điều chỉnh về 0 (nếu chưa, ta có thể điều chỉnh ma trận kernel).

$$\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \quad \Longrightarrow \quad \frac{1}{n}\sum_{i=1}^{n}\varphi(\mathbf{x}_i) = 0$$

Trong không gian Hilbert H, phép chiếu vuông góc từ một điểm dữ liệu $\mathbf{x}$ lên một hướng f có thể được thực hiện bằng cách tính toán giá trị của hàm $h_f(\mathbf{x})$.

$$h_{\mathbf{w}}(\mathbf{x}) = x\frac{\mathbf{w}}{||\mathbf{w}||} \quad \Longrightarrow \quad h_f(\mathbf{x}) = \left\langle \varphi(\mathbf{x}), \frac{f}{||f||_H} \right\rangle_H$$

Hàm này trả về một số thực, thể hiện khoảng cách từ điểm dữ liệu $\mathbf{x}$ đến mặt phẳng được tạo bởi hướng f trong không gian Hilbert H.

Khi áp dụng phép chiếu vuông góc h_f lên dữ liệu, ta có thể tính được phương sai thực nghiệm được thu thập bởi hàm h_f:

$$\hat{\text{var}}(h_{\mathbf{w}}) = \frac{1}{n}\sum_{i=1}^{n}\frac{(\mathbf{x}i^{\mathsf{T}}\mathbf{w})^2}{|\mathbf{w}|^2} \quad \Longrightarrow \quad \hat{\text{var}}(h\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\frac{\langle\varphi(\mathbf{x}_i), f\rangle_H^2}{|f|_H^2}$$

Trong PCA, các hướng chính là các vector eigenvector tương ứng với các giá trị riêng lớn nhất của ma trận hiệp phương sai của dữ liệu. Cụ thể, hướng chính thứ i được tính bằng cách lấy vector riêng tương ứng với giá trị riêng lớn thứ i của ma trận hiệp phương sai.

$$f_i = \underset{f \perp f_1,\ldots,f_{i-1}}{\arg\max} \ \hat{\text{var}}(h_f) \quad \text{s.t.} \quad |f|_H = 1$$

$$f_i = \underset{f \perp f_1,\ldots,f_{i-1}}{argmax} \ \sum_{i=1}^{n} f(\mathbf{x}_i)^2 \quad s.t \quad ||f||_H = 1$$

Kiểm tra tính hợp lệ của Kernel PCA với kernel tuyến tính. Chúng ta sử dụng kernel tuyến tính $K(x, y) = x^{\mathsf{T}}y$. Trong trường hợp này, ta có thể nhận thấy rằng không gian RKHS trở thành không gian các hàm tuyến tính dựa trên vectơ w với $f_w(x) = w^{\mathsf{T}}x$ và có dạng chuẩn hóa $||f_w||_H = ||w||_{\mathbb{R}^d}$. Chúng ta viết lại biểu thức phương sai hàm h_f như sau:

$$\hat{var}(h_{\mathbf{w}}) = \frac{1}{n}\sum_{i=1}^{n}\frac{(\mathbf{x}_i^{\mathsf{T}}\mathbf{w})^2}{|\mathbf{w}|^2} = \frac{1}{n|f\mathbf{w}|^2}\sum_{i=1}^{n} f_{\mathbf{w}}(\mathbf{x}_i)^2$$

Mối quan hệ tương đương giữa các vector w và w' tương ứng các hàm f_w và f'_w. Nếu $(w \perp w')$ nghĩa là tích vô hướng cả 2 bằng 0, thì điều này tương tự hai hàm f_w và f'_w cũng vuông góc với nhau $(f_w \perp f''_w)$ và có tích bằng 0. Điều này cho thấy mối quan hệ tương quan giữa các vector và hàm trong không gian RKHS trong quá trình Kernel PCA.

Định lý Representer khẳng định rằng các hàm trong có thể được biểu diễn dưới dạng tổ hợp tuyến tính của các hàm nhân tương ứng với các điểm dữ liệu. Trong trường hợp kernel tuyến tính, định lý vẫn có hiệu lực với kernel tuyến tính.

Kiểm chứng với Representer với i = 1,...,d được sắp xếp theo thứ tự giảm dần của giá trị riêng tương ứng. Chúng ta có:

$$\forall \mathbf{x} \in X, \quad f_i(\mathbf{x}) = \sum_{j=1}^{n}\alpha_{ij}K(\mathbf{x}_j, \mathbf{x})$$

với $\alpha_i = \begin{pmatrix}\alpha_{i,1} & \alpha_{i,2} \vdots \alpha_{i,n}\end{pmatrix} \in \mathbb{R}^n$

Vì thế chúng ta có biểu thức như sau:

$$||f||_H^2 = \sum_{k,l=1}^{n} \alpha_{i,k}\alpha_{i,l}K(\mathbf{x}_k, \mathbf{x}_l)$$

Tương tự cho:

$$\sum_{k=1}^{n} f_i(x_k)^2 = \alpha_i^T K^2 \alpha_i \quad \text{và} \quad \langle f_i, f_j \rangle_H = \alpha_i^T K \alpha_j$$

Kernel PCA tối đa hóa hàm trong α :

$$\alpha_i = \underset{\alpha \in \mathbb{R}^n}{\text{argmax}} \alpha^T K^2 \alpha$$

Việc này có nghĩa là chúng ta tìm các vector α_i để hàm $\alpha^T K^2 \alpha$ đạt giá trị lớn nhất. Các vector α_i này sẽ được sử dụng để xác định các thành phần chính trong quá trình Kernel PCA và giúp giảm chiều dữ liệu phi tuyến tính với các điều kiện ràng buộc:

$$\begin{cases} \alpha_i^T K \alpha_j = 0 \quad \text{for } j = 1, 2, ..., i-1 \\ \alpha_i^T K \alpha_j = 1 \end{cases}$$

Thực hiện tính toán các vector α_i, ta cần thực hiện phân rã trị riêng của ma trận kernel $K = U\Delta U^T$, trong đó $\Delta = \text{diag}(\Delta_1, \ldots, \Delta_n)$ là ma trận đường chéo chứa các trị riêng $\Delta_1 \geq \ldots \geq \Delta_n \geq 0$ và U là ma trận chứa các vector riêng tương ứng.

Gọi $K^{1/2} = U\Delta^{1/2}U^T$. Sau khi biến đổi biến $\beta = K^{1/2}\alpha$ ta có được biểu thức:

$$\beta_i = \arg\max_{\beta \in \mathbb{R}^n} \beta^T K \beta$$

Với điều kiện ràng buộcbuộc:

$$\begin{cases} \beta_i^T \beta_j = 0 \quad \text{for } j = 1, 2, ..., i-1 \\ \beta_i^T \beta_j = 1 \end{cases}$$

Khi đó, ta có $\beta_i = u_i$ là một giải pháp.Ngoài ra, để tính toán các vector α_i, ta có thể sử dụng công thức $\alpha_i = \sqrt{\frac{1}{\Delta_i}} u_i$. Việc này có nghĩa là để tìm giá trị α_i tương ứng với vector riêng u_i, ta chỉ cần chia mỗi phần tử của vector riêng u_i cho căn bậc hai của giá trị riêng tương ứng Δ_i.

Kernel PCA là một phương pháp giảm chiều dữ liệu phi tuyến tính. Quá trình Kernel PCA bao gồm các bước sau:

Tâm điểm ma trận Gram: Áp dụng hàm kernel lên các cặp điểm dữ liệu để tạo ma trận Gram $K_{ij} = \text{kernel}(x_i, x_j)$

Tính các vector riêng đầu tiên: Tìm các vector và trị riêng tương ứng của ma trận Gram $K = U\Delta U^T$. Sắp xếp các trị riêng theo thứ tự giảm dần.

Chuẩn hóa các vector riêng: $\alpha_i = \frac{\mathbf{u}_i}{\sqrt{\Delta_i}}$ Bước này đảm bảo các vector riêng có độ dài bằng một.

Phép chiếu của các điểm dữ liệu lên vector riêng thứ i được cho bởi tích ma trận giữa ma trận Gram K và vector α_i, tức là: $\mathbf{K}\alpha_i$

Kernel PCA cho phép chúng ta khám phá và giảm chiều dữ liệu phi tuyến tính bằng cách xác định các thành phần chính trong không gian đặc trưng, dựa trên hàm kernel đã chọn.

2.4.3 KMeans

Phân cụm k-means là một phương pháp quan trọng trong việc chia nhỏ dữ liệu thành các nhóm dựa trên sự tương đồng giữa chúng nhằm chia n quan sát thành k cụm sao cho mỗi quan sát thuộc cụm có trung bình gần nhất.

Từ góc độ tối ưu hóa, phân cụm k-means được thực hiện bằng cách tối thiểu hóa hàm mục tiêu. Quá trình này thực hiện các bước lặp lại để giảm thiểu hàm chi phí. Trong mỗi bước, các trung tâm cụm được điều chỉnh để giảm thiểu hàm chi phí. Quá trình này tiếp tục lặp lại cho đến khi hàm chi phí không thay đổi đáng kể hoặc đạt được một tiêu chuẩn dừng được xác định trước. Cụ thể, ta có:

Cho các điểm dữ liệu $x_1, ..., x_n$ trong không gian $\mathbb{R}^p$. Ta cần tìm các trung tâm u_j trong không gian $\mathbb{R}^p$ sao cho tổng bình phương khoảng cách giữa mỗi điểm dữ liệu x_i và trung tâm u_{s_i} được giảm thiểu. Điều này có thể được biểu diễn như sau:

$$\underset{\substack{\mu_j \in \mathbb{R}^p \\ s_i \in 1,\dots,k,}}{min} \underset{\substack{for \\ for}}{} \underset{\substack{j=1,\dots,k \\ i=1,\dots,n}}{} \sum_{i=1}^{n} ||\mathbf{x}_i - \mu_{\mathbf{s}_i}||_2^2$$

Trong công thức trên, ta tìm các giá trị tối thiểu cho tổng bình phương khoảng cách giữa mỗi điểm dữ liệu x_i và trung tâm u_{s_i}. Điều này được lặp lại cho mỗi giá trị của j từ 1 đến k, mỗi giá trị của s_i từ 1 đến k, và mỗi giá trị của i từ 1 đến n. K-means xen kẽ giữa hai bước:

Bước 1: cluster assignment: (phân cụm).

Cho cố định $u_1, ..., u_k$ gán mỗi x_i cho trọng tâm gần nó nhất.

$$\forall i, s_i \in \underset{s \in \{1,2,\dots,k\}}{\operatorname{argmin}} \|x_i - u_s\|_2^2$$

Gán mỗi quan sát s_i vào cụm có trung tâm gần nhất. Khoảng cách được tính dựa trên khoảng cách Euclid giữa quan sát và trung tâm cụm.

Bước 2: centroids update(cập nhật các điểm trung tâm).

Cập nhật trung tâm cụm bằng cách tính trung bình của tất cả các quan sát trong cụm. Với $s_1, ..., s_n$ là các phân bố được gán cho các điểm dữ liệu tương ứng $x_1, ..., x_n$. Cập nhật các điểm trung tâm như sau:

$$\forall j, \quad u_j \in \underset{u \in \mathbb{R}^p}{\operatorname{argmin}} \sum_{i:s_i=j} \|x_i - u\|_2^2$$

Khi đó u_j được gọi là điểm trung tâm của S_i và được viết lại như sau:

$$\forall j, \quad u_j = \frac{1}{|C_j|} \sum_{i \in C_j} x_i \quad \text{with} \quad C_j = \{i : s_i = j\}$$

công thức tính trung bình cộng của các điểm dữ liệu trong mỗi cụm. Trong đó: $_i$ là trung bình của cụm i. C_i là tập hợp các điểm dữ liệu thuộc cụm i. $|C_i|$ là số lượng điểm dữ liệu trong cụm i. x_i là một điểm dữ liệu trong cụm i. Công thức trên tính giá trị trung bình của các điểm dữ liệu trong cụm i bằng cách lấy tổng của các điểm dữ liệu trong cụm đó và chia cho số lượng điểm dữ liệu trong cụm. Kết quả là giá trị trung bình của các điểm dữ liệu trong cụm i. K-means giả định rằng các cụm là các hình cầu có bán kính đồng nhất và dữ liệu được phân tách bằng các cách biên tương đối thẳng. Trong khi đó, dữ liệu phi tuyến thường có các cấu trúc không đều và các cụm không có hình cầu. Để xử lý dữ liệu phi tuyến, có thể sử dụng các biến thể của K-means như Kernel K-means hoặc Spectral Clustering.

2.4.4 Kernel KMeans

Bây giờ chúng ta có thể điều chỉnh mục tiêu để hoạt động trong một không gian Hilbert tái sinh hạt nhân (RKHS). Cho các điểm dữ liệu $x_1, ..., x_n$ trong không gian X và một hạt nhân $K : X \times X \to \mathbb{R}$ xác định dương với H là RKHS tương ứng của nó, trong trường hợp này sẽ trở thành

$$u \underset{\substack{\mathbf{j} \in \mathbb{H} \\ s_i \in 1,\dots,k,}}{min} \underset{\substack{for \\ for}}{} \underset{\substack{j=1,\dots,k \\ i=1,\dots,n}}{} \sum_{i=1}^{n} ||\varphi(x_i) - \mu_{\mathbf{s}_i}||_2^2$$

Trọng tâm của φ_i được tính bằng công thức $\varphi_i = \frac{1}{n} \sum_{i=1}^{n} \varphi(x_i)$. Trong bài toán tối ưu này, chúng ta tìm một điểm u trong không gian Hilbert H sao cho tối ưu như sau:

$$\min_{u \in H} \sum_i ||\varphi(x_i) - u||_H^2$$

đây là giá trị tối thiểu của tổng bình phương khoảng cách giữa các điểm dữ liệu đã được biến đổi $\varphi(x_i)$ và trung tâm khối lượng trong không gian Hilbert H.

Để tìm trung tâm khối lượng φ_n, ta tính trung bình của các điểm dữ liệu đã được biến đổi $\varphi(x_i)$ qua tất cả các điểm dữ liệu x_i sử dụng hàm kernel φ và không gian Hilbert H. Bằng cách tìm trung tâm khối lượng $\mu = \varphi_n$, chúng ta có thể tìm được điểm u tối ưu trong không gian Hilbert H mà giảm thiểu tổng bình phương khoảng cách đến các điểm dữ liệu đã được biến đổi.

$$\begin{aligned}\sum_{i=1}^{n} |\varphi(x_i) - u_{s_i}|_H^2 &= \sum_{i=1}^{n} |\varphi(x_i)|_H^2 - \frac{2}{n} \left\langle \sum_{i=1}^{n} \varphi(x_i), u \right\rangle_H + ||u||_H^2 \\ &= \sum_{i=1}^{n} |\varphi(x_i)|_H^2 - 2 \langle \varphi(x_i), u \rangle_H + ||u||_H^2 \\ &= \sum_{i=1}^{n} |\varphi(x_i)|_H^2 - ||\varphi_n||_H^2 + ||\varphi_n - u||_H^2\end{aligned}$$

Khi u được đặt bằng trung tâm khối lượng φ_n, tức là $u = \frac{1}{n}\sum_{i=1}^{n}(\varphi(x_i))$, tổng bình phương khoảng cách giữa các điểm dữ liệu đã được biến đổi $\varphi(x_i)$ và u sẽ đạt giá trị tối thiểu. Quá trình lặp này tiếp tục cho đến khi không có sự thay đổi đáng kể trong giá trị của u hoặc đạt được điều kiện dừng khác nào đó. Khi thuật toán kết thúc, giá trị u tối ưu được tìm thấy và $u = \varphi(x_i)$ cho các điểm dữ liệu được gán với giá trị u tối ưu đó.

K-means và Kernel K-means đều có hai bước chính là cập nhật tâm cụm và phân cụm. Tuy nhiên, thứ tự thực hiện của hai bước này lại ngược nhau giữa hai phương pháp.

Trong K-means clustering, thực hiện bước phân cụm, trong đó tính toán khoảng cách Euclidean giữa các điểm dữ liệu và các tâm cụm để gán điểm vào các cụm tương ứng. Sau đó tiến hành bước cập nhật tâm bằng cách tính trung bình của các điểm trong cùng một cụm, từ đó xác định tâm mới.

Trong Kernel K-means clustering, chúng ta làm việc trong không gian Hilbert tái sinh hạt nhân (RKHS) bằng cách sử dụng phiên bản kernel của dữ liệu. Vì vậy, việc tính toán khoảng cách Euclidean trực tiếp không khả thi. Thay vào đó, chúng ta ánh xạ dữ liệu vào không gian RKHS thông qua hàm kernel. Trong không gian này, bước cập nhật tâm trở thành việc tính trung bình của các phiên bản kernel của các điểm trong cùng một cụm. Sau đó, bước phân cụm được thực hiện bằng cách gán các điểm dữ liệu vào cụm có tâm gần nhất dựa trên khoảng cách giữa các phiên bản kernel của chúng.

Do đó, thứ tự thực hiện của hai bước trong Kernel K-means clustering là ngược lại so với K-means clustering. Đầu tiên, chúng ta cập nhật tâm cụm bằng cách tính trung bình các phiên bản kernel của các điểm trong cùng một cụm. Sau đó, chúng ta thực hiện bước phân cụm bằng cách gán các điểm dữ liệu vào cụm có tâm gần nhất dựa trên khoảng cách giữa các phiên bản kernel của chúng.

2.4.5 Greedy algorithm

Greedy algorithm là một phương pháp tìm kiếm tối ưu đơn giản và có hiệu quả trong nhiều bài toán. Trong K-means clustering, greedy algorithm có thể được áp dụng để tìm kiếm các trung tâm cụm tối ưu.

Với 2 bước chính là cập nhật tâm điểm và phân cụm, chúng ta thực hiện xen kẽ như ở Kernel KMeans. Tuy nhiên có 1 vài thay đổi nhỏ khi áp dụng Greedy như sau:
Bước 1: centroids update Dựa trên các phân bố trước đó $s_1, ..., s_n$, cập nhật các trung tâm cụm

$$\forall j, \quad u_j \in \underset{u \in \mathbb{H}}{\operatorname{argmin}} \sum_{i:s_i=j} \|\varphi(x_i) - u\|_H^2$$

Tương đương với biểu thức:

$$\forall j, \quad u_j = \frac{1}{|C_j|} \sum_{i \in C_j} \varphi(x_i) \quad \text{with} \quad C_j = \{i : s_i = j\}$$

Bước 2: clusustering Với $_1, ..., _k$ cố định, gán từng điểm dữ liệu x_i vào trung tâm gần nhất:

$$\forall j, \quad u_j \in \underset{S \in 1,..,k}{\operatorname{argmin}} ||\varphi(x_i) - u_s||_H^2$$

Tương ứng:

$$\forall j, \quad u_j \in \underset{S \in 1,..,k}{\operatorname{argmin}} ||\varphi(x_i) - \frac{1}{|C_j|} \sum_{j \in C_s} \varphi(x_j)||_H^2$$

Chúng ta có các trung tâm cố định $_1, _2, ..., _k$ được thu được từ bước trước. Đối với mỗi điểm dữ liệu x_i, chúng ta tính khoảng cách bình phương giữa biểu diễn được chuyển đổi bằng kernel

$$\varphi(x_i)$$

và trung bình của các biểu diễn được chuyển đổi bằng kernel của các điểm dữ liệu trong cùng một cụm C_s mà nó có thể thuộc về.Việc gán được xác định bằng cách tìm giá trị của $s = 1, 2, ..., k$ làm giảm thiểu khoảng cách bình phương.

Tìm cụm s tối ưu cho x_i. Nó được tính toán bằng cách so sánh các giá trị khoảng cách giữa x_i và các trung tâm cụm $u_1, ..., u_k$ sử dụng kernel $K(x_i, x_j)$.

$$\forall j, \quad u_j \in \underset{S \in 1,..,k}{\operatorname{argmin}}(K(x_i, x_i) - \frac{2}{|C_s|} \sum_{j \in C_s} (K(x_i, x_j) + \frac{1}{|C_s|^2} \sum_{j,l \in C_s} (K(x_j, x_l))$$

Lưu ý rằng tất cả các phép toán được thực hiện bằng cách xử lý giá trị hạt nhân $K(x_i, x_j)$ duy nhất. Ước lượng không rõ ràng rằng chúng ta đang tối ưu hóa thực tế. Điều này đồng nghĩa với việc chúng ta không cần biết thông tin chi tiết về các điểm dữ liệu x_i và x_j, mà chỉ cần sử dụng giá trị hạt nhân để tính toán các khoảng cách và đánh giá sự tương đồng giữa chúng.

$$\min_{u \in H} \sum_{\text{for } i=1,2,..,n}^{n} ||\varphi(x_i) - \frac{1}{|C_s|} \sum_{j,l \in C_s} \varphi(x_i)||_H^2$$

Tương đương

$$\underset{\substack{\mu_j \in \mathbb{R}^p \\ s_i \in 1,...,k,}}{\underset{for}{min}} \underset{\substack{for \\ i=1,...,n}}{\underset{for}{} \underset{j=1,...,k}{}} (K(x_i, x_i) - \frac{2}{|C_{s_i}|} \sum_{j \in C_{s_i}} (K(x_i, x_j) + \frac{1}{|C_{s_i}|^2} \sum_{j,l \in C_s} (K(x_j, x_l))$$

Trong đó

$$\sum_{i=1}^{n} \frac{1}{|C_{s_i}|^2} \sum_{j,l \in C_s} (K(x_j, x_l)) = \sum_{i=1}^{n} \frac{1}{|C_l|} \sum_{j,l \in C_l} (K(x_i, x_j))$$

2 biểu thức này tương đương nhau Biểu thức đầu sử dụng biến đổi phiên bản kernel của các điểm dữ liệu $\varphi(x_i)$ để tính toán khoảng cách giữa các phiên bản kernel. Ở biểu thức thứ hai, ta sử dụng hạt nhân $K(x_i, x_j)$ để tính toán khoảng cách giữa các điểm dữ liệu xi và xj trong không gian Hilbert tái sinh hạt nhân (RKHS). Thay vì tính toán trực tiếp các vector $\varphi(x_i)$ và $\varphi(x_j)$, ta thay thế chúng bằng giá trị hạt nhân $K(x_i, x_j)$ tương ứng. Mặc dù biểu thức có sự khác biệt về cách tính toán, nhưng chúng tương đương với nhau trong việc xác định các phân cụm tối ưu. Việc tối thiểu hóa giá trị trong cả hai biểu thức đều nhằm mục đích tìm ra các phân cụm tối ưu cho dữ liệu. Từ đó ta có biểu thức sau:

$$\sum_{i=1}^{n} \frac{1}{|C_{s_i}|} \sum_{j \in C_{s_i}} (K(x_j, x_l)) = \sum_{l=1}^{n} \frac{1}{|C_l|} \sum_{j,l \in C_l} (K(x_i, x_j))$$

Tóm lại ,Bước đầu tiên, ta tính toán biểu diễn $\varphi(x_i)$ cho mỗi điểm xi trong không gian Hilbert tái sinh hạt nhân (RKHS). Tiếp theo, ta tìm các chỉ số s_i để gán mỗi điểm x_i vào trung tâm gần nhất. Điều này được thực hiện bằng cách tính toán tổng các khoảng cách bình phương giữa biểu diễn kernel của xi và trung tâm μi tương ứng. Chỉ số s_i được chọn là giá trị nhỏ nhất trong tất cả các giá trị này, đại diện cho trung tâm gần nhất. Sau đó, ta loại bỏ các thuật ngữ hằng số $K(x_i, x_i)$ (các giá trị kernel của x_i với chính nó) khỏi biểu thức mục tiêu. Kết quả là ta thu được biểu thức:

$$\max_{\substack{s_i \in 1,...,k \\ \text{for } i=1,2,..,n}} \frac{1}{|C_l|} \sum_{i,j \in C_l} K(x_i, x_j)$$

Điều này có nghĩa là ta đang tìm chỉ số s_i để tổng các giá trị kernel $K(xi, xj)$ trong cụm C_l đạt giá trị lớn nhất.

Bài toán này thuộc loại bài toán tối ưu tổ hợp khó. Nó thuộc lớp các bài toán NP-hard, có nghĩa là không có thuật toán hiệu quả để giải quyết nó trong thời gian đa thức. Trong trường hợp này, việc tìm ra tập chỉ số s_i tối ưu để tối thiểu hóa hoặc tối đa hóa hàm mục tiêu là một vấn đề khó và yêu cầu sử dụng các phương pháp

tìm kiếm và tối ưu hóa phức tạp như tìm kiếm heuristics, thuật toán tiến hóa, hoặc phương pháp tìm kiếm local.

Việc kết hợp Greedy Algorithm vào Kernel KMeans là để cải thiện hiệu suất và kết quả của thuật toán. Greedy Algorithm là một phương pháp tìm kiếm trực tiếp và tuần tự, nhằm tối ưu hóa mục tiêu của Kernel K-means có thể cải thiện thuật toán bằng cách tìm kiếm các trung tâm cụm tốt hơn và tối ưu hóa hàm mục tiêu thay vì sử dụng các trung tâm cụm ngẫu nhiên.

Trong 1 số trường hợp thì greedy algorithm sẽ không hoạt động tốt Local Optima: Greedy Algorithm có thể bị mắc kẹt trong các cụm tối ưu cục bộ và không tìm được cụm toàn cục tối ưu. Điều này xảy ra khi quá trình tối ưu hóa dẫn đến một phân bố sai lệch của các trung tâm cụm ban đầu.

Không đồng nhất: Khi dữ liệu không đồng nhất, tức là các cụm có kích thước và mật độ khác nhau, Greedy Algorithm có thể không phân cụm chính xác các điểm dữ liệu vào các cụm phù hợp. Nhiễu: Nếu dữ liệu chứa nhiễu, tức là các điểm dữ liệu không thuộc vào bất kỳ cụm nào hoặc làm cho phân cụm trở nên mơ hồ, Greedy Algorithm có thể không xử lý nhiễu hiệu quả và cho kết quả không tốt.

Ngoài Greedy Algorithm, còn có các phương pháp khác để tối ưu hóa Kernel K-means, chẳng hạn như phương pháp Spectral Clustering, phương pháp Relaxed Kernel K-means, hoặc phương pháp tối ưu hóa sử dụng các thuật toán tối ưu hóa khác nhau như Gradient Descent. Các phương pháp này có thể mang lại kết quả tốt hơn trong một số trường hợp

2.4.6 Spectral clustering

Để giải quyết các trường hợp mà Greedy algorithm không đạt hiểu quả mong muốn. Spectral clustering algorithm được đề xuất để giải quyết vấn đề này.

Spectral clustering algorithms là một nhóm các thuật toán phân cụm dựa trên việc sử dụng phân rã giá trị riêng và vector riêng của ma trận tương đồng hoặc ma trận Laplace để thực hiện việc phân cụm dữ liệu.

Đầu tiên, ta xem xét biểu thức:

$$\max_{\substack{s_i \in 1,\dots,k \\ \text{for } i=1,2,..,n}} \frac{1}{|C_l|} \sum_{i,j \in C_l} K(x_i, x_j)$$

Và ta có các thông tin sau:

1. Ma trận gán nhị phân A có giá trị trong 0, 1 và có kích thước n × k, trong đó tổng các hàng của ma trận A bằng một. Ma trận A được sử dụng để gán mỗi điểm dữ liệu vào một trong k cụm. 2. Ma trận điều chỉnh đường chéo D kích thước k×k với các phần tử trên đường chéo $D_{j,j}$ bằng $(\sum_{i=1}^{n} A_{i,j})^1$: nghịch đảo của kích thước của cụm j. Ma trận D được sử dụng để điều chỉnh tầm quan trọng của các cụm trong quá trình tối ưu hóa.

Mục tiêu tối ưu là

$$\max_{A,D} \text{trace}(D^{1/2} A^T K A D^{1/2}) \quad \text{s.t.} \quad (1) \quad \text{and} \quad (2)$$

với các ràng buộc 1 và 2. Điều kiện ràng buộc trên A, D là $D_{1/2} A^T A D^{1/2} = I$. Một phương pháp thả lỏng tự nhiên là bỏ các ràng buộc (1, 2) trên A và D. Đặt $Z = AD^{1/2}$ để tối ưu hóa

Một giải pháp Z^* cho bài toán này có thể được thu được bằng cách tính toán các vector riêng của K tương ứng với k giá trị riêng lớn nhất. Thủ tục này có liên quan đến thuật toán PCA trong không gian kernel.

Để đạt được nghiệm gần đúng (A, D) của bài toán ban đầu từ nghiệm chính xác của bài toán thoải mái Z^*

Với các ràng buộc ban đầu trên ma trận A, mỗi hàng của A có một phần tử không bằng không duy nhất. Ta có thể tính toán giá trị lớn nhất của mỗi hàng của ma trận Z^*.

Chuẩn hóa các hàng của Z* để có độ dài 2-norm đơn vị, tức là chia mỗi hàng cho độ dài 2-norm của nó.

Áp dụng thuật toán K-means truyền thống trên các hàng của Z^* đã được chuẩn hóa. Quá trình này gọi là spectral clustering.

Một phương pháp khác là chọn một biến thể của các bước trước đó.

Thứ tự thực hiện như sau:

Cập nhật điểm trung tâm:

Bước 1: Tạo ma trận kernel từ dữ liệu ban đầu sử dụng kernel function.

Bước 2: Từ ma trận kernel, tính ma trận Laplacian của đồ thị tương ứng.

Bước 3: Tính toán các giá trị riêng và vector riêng tương ứng của ma trận Laplacian.

Bước 4: Chọn k giá trị riêng lớn nhất và tạo ma trận Z từ các vector riêng tương ứng.

Bước 5: Áp dụng thuật toán K-means trên các hàng của ma trận Z để cập nhật các điểm trung tâm cụm.

Tìm kiếm điểm:

Bước 1: Dựa vào các điểm trung tâm cụm đã được cập nhật, tính toán khoảng cách giữa mỗi điểm dữ liệu và các điểm trung tâm.
Bước 2: Gán nhãn cho mỗi điểm dữ liệu bằng cách chọn điểm trung tâm gần nhất.
Bước 3: Cập nhật lại các điểm trung tâm cụm bằng cách tính toán trung bình của các điểm dữ liệu trong cùng một cụm.
Bước 4: Lặp lại các bước trên cho đến khi không có sự thay đổi đáng kể trong việc gán nhãn và cập nhật điểm trung tâm.

2.4.7 Canonical Correlation Analysis

Phân tích tương quan canon (Canonical Correlation Analysis - CCA) nhằm xác định và đo lường mối quan hệ giữa hai tập biến số. Nó tập trung vào tương quan giữa các tổ hợp tuyến tính của các biến trong hai tập dữ liệu. Ý tưởng của CCA là tìm cặp tổ hợp tuyến tính có mối tương quan lớn nhất và sau đó tiếp tục tìm các cặp khác không tương quan với nhau. Các cặp tổ hợp tuyến tính này được gọi là các biến canon và mối tương quan của chúng được gọi là các tương quan canon. Các tương quan canon đo lường sức mạnh của mối quan hệ giữa hai tập biến số. Mục tiêu của CCA là tập trung mối quan hệ đa chiều giữa hai tập biến số vào một số ít cặp biến canon.

Cho hai ma trận $X = [x_1, ..., x_n]$ kích thước R^{pn} và $Y = [y_1, ..., y_n]$ kích thước R^{dn} đại diện cho hai góc nhìn (views) khác nhau của cùng một tập dữ liệu, mục tiêu của phân tích tương quan cổ điển (CCA) là tìm ra các cặp hướng trong hai góc nhìn đó có mối tương quan tối đa. Chúng ta tìm kiếm các hướng (directions) trong không gian p và d chiều (cho Y) sao cho mối tương quan giữa các hướng tương ứng là lớn nhất.

Giả sử rằng các tập dữ liệu đã được điều chỉnh về trung tâm, chúng ta muốn tối đa hóa...

$$\max_{w_a \in R^p, w_b \in R^d} \frac{\frac{1}{n}\sum_{i=1}^{n} W_a^T x_i y_i^T W_b}{(\frac{1}{n}\sum_{i=1}^{n} W_a^T x_i x_i^T W_a)^{1/2}(\frac{1}{n}\sum_{i=1}^{n} W_b^T y_i y_i^T W_b)^{1/2})}$$

Giả sử rằng các cặp (x_i, y_i) là các mẫu độc lập và có phân phối không xác định, CCA tìm cách tối đa hóa giá trị tương quan giữa hai tập dữ liệu X và Y.

$$\max_{w_a \in \mathbb{R}^p, w_b \in \mathbb{R}^d} \frac{\text{cov}(W_a^T X, W_b^T Y)}{\sqrt{\text{var}(W_a^T X) \cdot \text{var}(W_b^T Y)}}$$

Sau khi tìm được cặp đầu tiên của các hướng canonical correlation, ta có thể tiếp tục tìm các hướng canonical tiếp theo bằng cách giải một vấn đề tương tự. Tuy nhiên, trong quá trình này, ta áp đặt ràng buộc là các hướng canonical tiếp theo phải vuông góc với các hướng canonical đã được tìm trước đó. Cách tiếp cận này giúp tìm ra các hướng canonical tối ưu khác nhau và đảm bảo tính đa dạng của các hướng tìm được.

Quá trình này có thể được lặp lại cho đến khi ta thu được số lượng hướng canonical mong muốn hoặc khi không còn sự cải thiện đáng kể trong giá trị tương quan giữa các dữ liệu. Việc lặp lại này giúp tìm ra các hướng canonical tiếp theo một cách tuần tự và đồng thời đảm bảo tính chất vuông góc giữa chúng.

$$\max_{w_a \in \mathbb{R}^p, w_b \in \mathbb{R}^d} \frac{(W_a^T X_i Y_i^T W_b)}{(W_a^T X_i X_i^T W_a)^{1/2} \cdot (W_b^T Y_i Y_i^T W_b)^{1/2}}$$

Sau khi loại bỏ tính không xác định về tỉ lệ, ta có biểu thức như sau:

$$\max_{w_a \in \mathbb{R}^p, w_b \in \mathbb{R}^d} (W_a^T X_i Y_i^T W_b) \quad \text{s.t.} \quad (W_a^T X_i X_i^T W_a) = 1 \quad \text{s.t.} \quad (W_b^T Y_i Y_i^T W_b) = 1$$

Công thức này tối ưu hóa độ tương quan giữa X và Y thông qua việc tìm các hướng W_a và W_b tối ưu. Các ràng buộc đảm bảo tính đơn vị của các vectơ hướng để đảm bảo tính chuẩn xác và ổn định của giải thuật. Bằng cách tối ưu công thức này, ta có thể tìm được các hướng tương quan tối ưu giữa hai tập dữ liệu X và Y.

Tiếp đó, tồn tại λ_a và λ_b là các tham số Lagrange để điều chỉnh tầm quan trọng của các ràng buộc sao cho

$$\min_{w_a \in \mathbb{R}^p, w_b \in \mathbb{R}^d} -W_a^T X_i Y_i^T W_b + \frac{\lambda_a}{2}(W_a^T X_i X_i^T W_a - 1) + \frac{\lambda_b}{2}(W_b^T Y_i Y_i^T W_b - 1)$$

Vấn đề này tối ưu hóa sự tương quan giữa X và Y và đồng thời đảm bảo tính đơn vị của các vectơ hướng w_a và w_b thông qua các ràng buộc bổ sung. Tìm được các hướng tương quan tối ưu giữa hai tập dữ liệu X và Y.

Bằng cách lấy đạo hàm và đặt gradient bằng không, chúng ta có thể suy ra các phương trình sau:

$$-X^T Y w_b + \lambda_a X^T X w_a = 0$$
$$-Y^T X w_a + \lambda_b Y^T Y w_b = 0$$

Bằng cách nhân phương trình thứ nhất với w_a^T và phương trình thứ hai với w_b^T, sau đó trừ hai phương trình này, ta được:

$$\lambda_a w_a^T X^T X w_a = \lambda_b w_b^T Y^T Y w_b = \lambda_a = \lambda_b = \lambda$$

Điều này ngụ ý rằng λ_a và λ_b bằng nhau và có thể được ký hiệu bằng λ. Và sau đó, chúng ta thu được bài toán giá trị riêng tổng quát.

$$\begin{bmatrix} 0 & X^T Y \\ X^T Y & 0 \end{bmatrix} \begin{bmatrix} w_a \\ w_a \end{bmatrix} = \lambda \begin{bmatrix} X^T Y & 0 \\ 0 & X^T Y \end{bmatrix} \begin{bmatrix} w_a \\ w_a \end{bmatrix}$$

Ta định nghĩa như sau:

Giả sử các ma trận hiệp phương sai là khả nghịch, bài toán giá trị riêng tổng quát tương đương với:

$$\sum B^{1/2} \sum A = \lambda \sum B^{1/2} w$$

Tương đương với giá trị riêng:

$$\sum B^{1/2} \sum A \sum B^{1/2} (\sum B^{1/2} w) = \lambda (\sum B^{1/2} w)$$

Kernel CCA: là một phương pháp phân tích tương quan tuyến tính giữa hai tập dữ liệu phi tuyến bằng cách sử dụng kernel.

Tương tự như Kernel PCA , 2 tập X, Y đều có thể được ánh xạ lên không gian RKHS. Ta có công thức CCA khi áp dụng kernel như sau:

$$\max_{f_a \in H_a, f_b \in H_b} \frac{\frac{1}{n} \sum_{i=1}^n \langle (f_a, \varphi_a(x_i)\rangle_H \langle (f_b, \varphi_b(x_i)\langle_H}{(\frac{1}{n} \sum_{i=1}^n \langle (f_a, \varphi_a(x_i)\rangle_H^2)^{1/2} (\frac{1}{n} \sum_{i=1}^n \langle (f_a, \varphi_a(x_i)\rangle_H^2)^{1/2}}$$

Tương tự:

$$\max_{f_a \in H_a, f_b \in H_b} \frac{\frac{1}{n} \sum_{i=1}^n f_a x_i f_b x_i}{(\frac{1}{n} \sum_{i=1}^n (f_a x_i)^2)^{1/2} (\frac{1}{n} \sum_{i=1}^n (f_b x_i)^2)^{1/2}}$$

Chúng ta có thể áp dụng định lý representer và tìm kiếm các giải pháp $f_a(.) = \sum_{i=1}^n [K_i \alpha]_i (x_i, .)$ và $f_b(.) = \sum_{i=1}^n [K_i \beta]_i (x_i, .)$. Chúng ta có được biểu thức cuối cùng như sau:

$$\max_{\alpha \in H_a, \beta \in H_b} \frac{\frac{1}{n} \sum_{i=1}^n [K_a \alpha]_i [K_b \beta]_i}{(\frac{1}{n} \sum_{i=1}^n [K_a \alpha]_i^2)^{1/2} (\frac{1}{n} \sum_{i=1}^n [K_b \beta]_i^2)^{1/2}}$$

Ta có thể viết lại biểu thức tương đương sau:

$$\max_{\alpha \in H_a, \beta \in H_b} \frac{\alpha^T K_a K_b \beta}{(\alpha^T K_a^2 \beta)^{1/2} (\alpha^T K_b^2 \beta)^{1/2}}$$

Sau khi loại bỏ sự không chắc chắn về tỉ lệ của α và β với α và β là các vector, và K_a và K_b là các ma trận. Ta có thể viết lại công thức tối ưu nhất như sau:

$$\max_{\alpha \in H_a, \beta \in H_b} \alpha^T K_a K_b \beta \quad \text{s.t.} \quad (\alpha^T K_a^2 \beta) = 1 \quad \text{s.t.} \quad (\alpha^T K_b^2 \beta) = 1$$

Điều này cũng dẫn đến một bài toán giá trị riêng tổng quát (generalized eigenvalue problem). Việc giải quyết bài toán giá trị riêng tổng quát là quan trọng để tìm các hướng canon bao gồm tìm kiếm các giá trị riêng và vector riêng sao cho một phép biến đổi tuyến tính giữa hai tập dữ liệu được tối ưu hóa. Tuy nhiên, để thu được các hướng canon tiếp theo, chúng ta cần đặt thêm ràng buộc về tính chéo giao giữa các hướng canon đã tìm được trước đó và các hướng canon mới để chắc rằng các hướng canon là độc lập và không tương quan với nhau. Điều này giúp chúng ta hiểu rõ hơn về mối quan hệ giữa các tập dữ liệu.

Tuy nhiên, chúng ta cần chú ý rằng nếu K_a và K_b là ma trận khả nghịch, ta có thể thực hiện một thay đổi biến số $\alpha' = K_a\alpha$ và $\beta' = K_a\beta$, từ đó thu được biểu thức tương đương.

Thay đổi biến số bằng cách đặt $\alpha' = K_a\alpha$ và $\beta' = K_b\beta$.

Biểu diễn lại biểu thức trên bằng cách sử dụng α' và β':

$$\max_{\alpha' \in H_a, \beta' \in H_b} (\alpha'^T \beta') \quad \text{s.t.} \quad (\alpha'^T \alpha') = 1 \quad \text{s.t.} \quad (\beta'^T \beta') = 1.$$

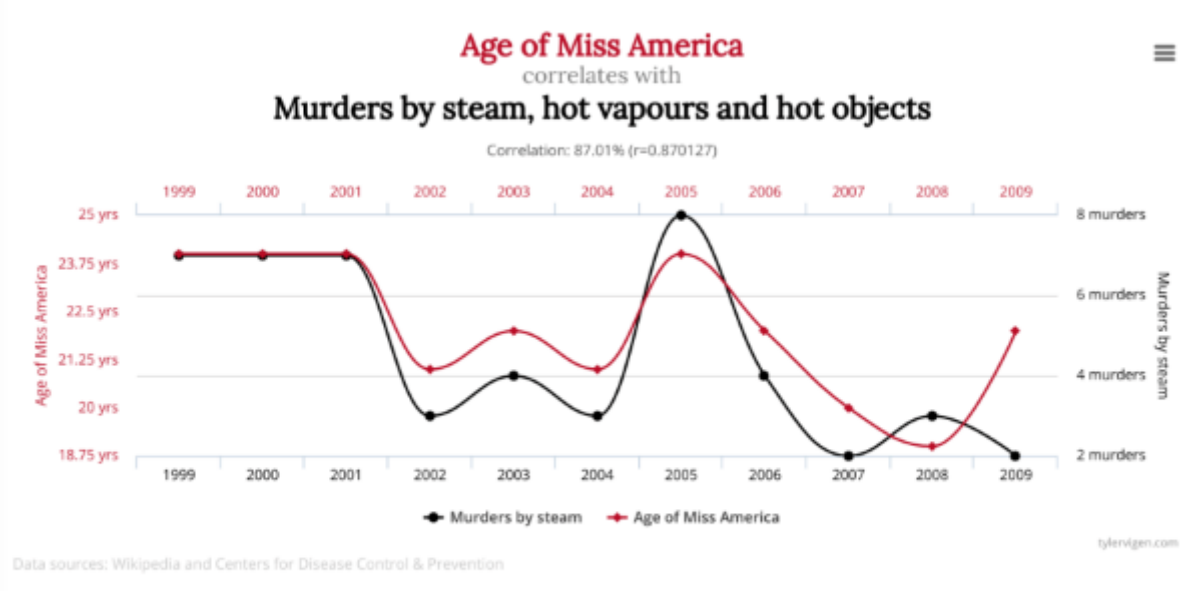


Figure: http://www.tylervigen.com/.

Khi làm việc trên không gian có số chiều cao hoặc vô hạn, việc tìm thấy các tương quan gây nhiễu là dễ dàng hơn, và điều này có thể dẫn đến xuất hiện các tương quan không có ý nghĩa thực tế hoặc ngẫu nhiên. Các tương quan gây nhiễu này cho kết quả không tốt.

1. Quá khớp dữ liệu (overfitting)
2. Không ổn định số học, do cần phải lấy nghịch đảo các ma trận kernel.

Để giải quyết cả hai vấn đề trên ta áp dụng biện pháp điều chuẩn (Regularize). Regularize là một kỹ thuật nhằm giảm thiểu overfitting (quá khớp) và cải thiện khả năng tổng quát hóa của mô hình. Cụ thể là một thuật toán như L1 regularization (điều chuẩn L1) hoặc L2 regularization (điều chuẩn L2) nhằm giảm độ phức tạp của mô hình và tránh việc quá tập trung vào các điểm dữ liệu riêng lẻ, từ đó giúp mô hình tổng quát hóa tốt hơn trên dữ liệu mới.

Đầu tiên, ta cần tìm các hướng $smooth(f_a, f_b)$ bằng cách giới hạn giá trị của $||f_a||_{H_a} v ||f_b||_{H_b}$.

Tiếp theo, thay thế các ràng buộc $\alpha^T K_a^2 \alpha = 1$ bằng:

$$(1-\gamma)\alpha^T K_a^2 \alpha + \gamma \underbrace{\alpha K_a^2 \alpha}_{||f_a||_H^2} = 1$$

và tương tự cho $\beta^T K_b^2 \beta = 1$

3 Tài liệu tham khảo

Tài liệu

[1] https://mva-kernel-methods.github.io/course-2021-2022/