MAT8034: Machine Learning

Kernel Methods

Fang Kong

https://fangkongx.github.io/Teaching/MAT8034/Spring2025/index.html

Part of slide credit: Stanford CS229

Outline

- Kernel methods
  - Feature maps
  - LMS (least mean squares) with features
  - LMS with the kernel trick
  - Properties of kernels

Feature maps

Feature maps

- In previous methods (linear regression)
  - We use $\theta^\top x = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ to predict the label

- What if the label can be more accurately represented as a non-linear function of x?
- Suppose the (new) feature is

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \end{bmatrix} \in \mathbb{R}^4$$

Feature maps

- Consider the cubic functions
  - $y = \theta^\top \phi(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$
- In this case, the objective can be viewed as a linear function over the variables $\phi(x)$
- For clarity
  - x: attributes
  - $\phi(x)$: features
  - ϕ: feature map

LMS with features

LMS

- Recall in linear regression, gradient descent gives

$$\begin{aligned}\theta &:= \theta + \alpha \sum_{i=1}^{n} \left(y^{(i)} - h_\theta(x^{(i)})\right) x^{(i)} \\ &:= \theta + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \theta^T x^{(i)}\right) x^{(i)}.\end{aligned}$$

- Similarly, when with feature maps

$$\theta := \theta + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \theta^T \phi(x^{(i)})\right) \phi(x^{(i)})$$

Disadvantages

- Computationally expensive
- let $\phi(x)$ be the vector that contains all the monomials of x with degree $\leq\ 3$
  - Dimension of $\phi(x)$: d^3
  - When $d = 1000, 10^9$
- *Can we avoid this?*

$$\phi(x) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_1^2 \\ x_1 x_2 \\ x_1 x_3 \\ \vdots \\ x_2 x_1 \\ \vdots \\ x_1^3 \\ x_1^2 x_2 \\ \vdots \end{bmatrix}$$

Disadvantages

- *Can we avoid this d^3 computation cost?*

- Though the unknown vector θ is also of this dimension

$$\phi(x) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_1^2 \\ x_1 x_2 \\ x_1 x_3 \\ \vdots \\ x_2 x_1 \\ \vdots \\ x_1^3 \\ x_1^2 x_2 \\ \vdots \end{bmatrix}$$

LMS with the kernel trick

Any great form of θ?

- With the GD, θ can be represented as a linear combination of the vectors $\phi(x)$
- By induction
  - At step 0, initialize $\theta = 0 = \sum_i 0 \cdot \phi(x^{(i)})$
  - Suppose some step, $\theta = \sum_i \beta_i \cdot \phi(x^{(i)})$
  - Then in the next step

$$
\begin{aligned}
\theta &:= \theta + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \theta^T \phi(x^{(i)})\right) \phi(x^{(i)}) \\
&= \sum_{i=1}^{n} \beta_i \phi(x^{(i)}) + \alpha \sum_{i=1}^{n} \left(y^{(i)} - \theta^T \phi(x^{(i)})\right) \phi(x^{(i)}) \\
&= \sum_{i=1}^{n} \underbrace{\left(\beta_i + \alpha \left(y^{(i)} - \theta^T \phi(x^{(i)})\right)\right)}_{\text{new } \beta_i} \phi(x^{(i)})
\end{aligned}
$$

Idea: represent θ by β

- Derive the update rule of β

$$\beta_i := \beta_i + \alpha \left(y^{(i)} - \theta^T \phi(x^{(i)})\right)$$

$$\theta = \textstyle\sum_{j=1}^{n} \beta_j \phi(x^{(j)})$$

$$\beta_i := \beta_i + \alpha \left(y^{(i)} - \sum_{j=1}^{n} \beta_j \phi(x^{(j)})^T \phi(x^{(i)})\right)$$

  - Denote the inner product of the two feature vectors as $\langle \phi(x^{(j)}), \phi(x^{(i)}) \rangle$

Can we accelerate computation?

- At each iteration, we need to compute $\langle \phi(x^{(j)}), \phi(x^{(i)}) \rangle, \forall j, i \in [n]$

- Acceleration
  - 1. It does not depend on iteration, we can compute it once before starts
  - 2. Computing the inner product does not necessarily require computing $\phi(x^{(i)})$ (see the next page)

Computing $\langle \phi(x^{(j)}), \phi(x^{(i)}) \rangle$

$$
\begin{aligned}
\langle \phi(x), \phi(z) \rangle &= 1 + \sum_{i=1}^{d} x_i z_i + \sum_{i,j \in \{1,\ldots,d\}} x_i x_j z_i z_j + \sum_{i,j,k \in \{1,\ldots,d\}} x_i x_j x_k z_i z_j z_k \\
&= 1 + \sum_{i=1}^{d} x_i z_i + \left(\sum_{i=1}^{d} x_i z_i \right)^2 + \left(\sum_{i=1}^{d} x_i z_i \right)^3 \\
&= 1 + \langle x, z \rangle + \langle x, z \rangle^2 + \langle x, z \rangle^3 \qquad (5.9)
\end{aligned}
$$

- Above all, the computation only requires $O(d)$

Kernel: definition

- Define the Kernel corresponding to the feature map φ as a function that maps $\mathcal{X} \times \mathcal{X} \to R$ satisfying

$$K(x, z) \triangleq \langle \phi(x), \phi(z) \rangle$$

The final algorithm

- Update β

1. Compute all the values $K(x^{(i)}, x^{(j)}) \triangleq \langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$ using equation ($\boxed{5.9}$) for all $i, j \in \{1, \ldots, n\}$. Set $\beta := 0$.

2. **Loop:**

$$\forall i \in \{1, \ldots, n\}, \beta_i := \beta_i + \alpha \left(y^{(i)} - \sum_{j=1}^{n} \beta_j K(x^{(i)}, x^{(j)}) \right) \tag{5.11}$$

Or in vector notation, letting K be the $n \times n$ matrix with $K_{ij} = K(x^{(i)}, x^{(j)})$, we have

$$\beta := \beta + \alpha(\vec{y} - K\beta)$$

- Compute the prediction

$$\theta^T \phi(x) = \sum_{i=1}^{n} \beta_i \phi(x^{(i)})^T \phi(x) = \sum_{i=1}^{n} \beta_i K(x^{(i)}, x)$$

Observation

- We do not need to know about the feature map, but only the kernel function

Properties of kernels

What kinds of kernels can correspond to some feature map?

- Or in other words, given a kernel function $K(\cdot,\cdot)$, can we tell if there is some feature mapping ϕ so that $K(x, z) = \langle \phi(x), \phi(z) \rangle$

- Let's consider some examples

Example 1: $K(x,z) = (x^T z)^2$

- Reduction:

$$\begin{aligned} K(x,z) &= \left(\sum_{i=1}^{d} x_i z_i\right)\left(\sum_{j=1}^{d} x_j z_j\right) \\ &= \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j z_i z_j \\ &= \sum_{i,j=1}^{d} (x_i x_j)(z_i z_j) \end{aligned}$$

- The feature mapping corresponds to

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix}$$

Example 2

- Consider
$$\begin{aligned} K(x,z) &= (x^T z + c)^2 \\ &= \sum_{i,j=1}^{d} (x_i x_j)(z_i z_j) + \sum_{i=1}^{d} (\sqrt{2c}x_i)(\sqrt{2c}z_i) + c^2. \end{aligned}$$

- The feature mapping corresponds to
  - The parameter c controls the relative weighting between first- and second-order terms

$$\phi(x) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \\ \sqrt{2c}x_1 \\ \sqrt{2c}x_2 \\ \sqrt{2c}x_3 \\ c \end{bmatrix}$$

Example 3: $K(x, z) = (x^T z + c)^k$

- Corresponding of all monomials of the form $x_{i_1}, x_{i_2}, \ldots$ that are up to order k

- Do not need to handle $O(d^k)$ computation, but only $O(d)$ for kernel function

Kernels as similarity metrics

- Different view of kernels from similarity
  - If $\phi(x)$ and $\phi(z)$ are close together, then expect $K(x, z)$ to be large
  - If $\phi(x)$ and $\phi(z)$ are far, expect $K(x, z)$ to be small

- The kernel can be regarded as some similarity measures
  - For example,

$$K(x, z) = \exp\left(-\frac{||x - z||^2}{2\sigma^2}\right)$$

  - Close to 1 if x and z are similar
  - Yes, this kernel is called the Gaussian kernel, and corresponds to some feature mappings

Necessary conditions for valid kernels

- What properties a kernel function satisfies?
  - 1. Symmetric

$$K_{ij} = K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)}) = \phi(x^{(j)})^T \phi(x^{(i)}) = K(x^{(j)}, x^{(i)}) = K_{ji}$$

  - 2. Positive semi-definite

$$\begin{aligned}
z^T K z &= \sum_i \sum_j z_i K_{ij} z_j \\
&= \sum_i \sum_j z_i \phi(x^{(i)})^T \phi(x^{(j)}) z_j \\
&= \sum_i \sum_j z_i \sum_k \phi_k(x^{(i)}) \phi_k(x^{(j)}) z_j \\
&= \sum_k \sum_i \sum_j z_i \phi_k(x^{(i)}) \phi_k(x^{(j)}) z_j \\
&= \sum_k \left(\sum_i z_i \phi_k(x^{(i)}) \right)^2 \\
&\geq 0.
\end{aligned}$$

Sufficient conditions for valid kernels

- The necessary conditions are also sufficient

Theorem (Mercer). Let $K : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ be given. Then for K to be a valid (Mercer) kernel, it is necessary and sufficient that for any $\{x^{(1)}, \ldots, x^{(n)}\}$, $(n < \infty)$, the corresponding kernel matrix is symmetric positive semi-definite.

Applications of kernel methods

- Image classification with objective to be strings
  - Each length-k substring in x can be regarded as features
  - 26^k substrings
  - The feature dimension: 26^k
  - Using kernel methods, the computational cost reduces to 26

- Kernel tricks:
  - Any learning algorithm that you can write in terms of only inner products <x,z> between input attribute vectors, then you can replace this with K(x, z) where K is a kernel

Summary

- Kernel methods
  - Feature maps
    - Non-linear features
  - LMS (least mean squares) with features
  - LMS with the kernel trick
    - θ is a linear combination of $\phi(x)$
    - Reduce the computational cost from $O(d^k)$ to $O(d)$
  - Properties of kernels
    - Symmetric, positive semi-definite