Discussion

He He

CDS, NYU

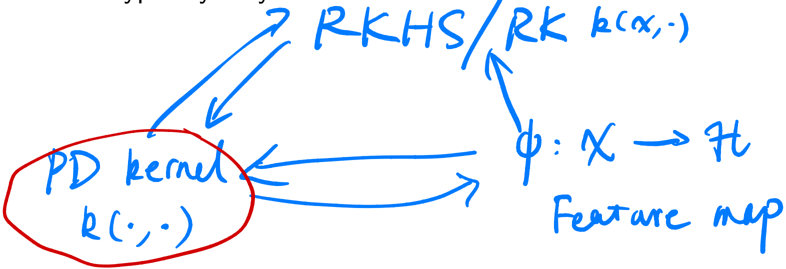
March 2, 2021

Motivation for kernels

- Our data is typically not linearly separable.
- But we like to work with linear models.
- Adding features (going to high-dimensional space) allow us to use linear models for complex data.
- Kernels allow us to think about similarities rather than feature engineering.

Two perspectives on kernels

- Given a feature map $\phi : \mathcal{X} \to \mathcal{H}$, we can define a kernel function $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$.
- Given a PD kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$, there exists a corresponding feature map.
  - Note that the kernel does not uniquely define the feature map.
- In practice we typically only work with the kernel function.



RBF Kernel

RBF Basis

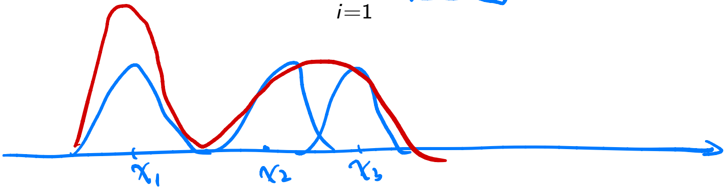
Input space $\mathcal{X} = R^d$

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

where σ^2 is known as the bandwidth parameter.

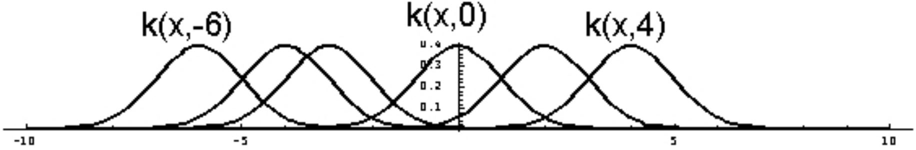
- Suppose we have 6 training examples: $x_i \in \{-6, -4, -3, 0, 2, 4\}$.
- If representer theorem applies, then

$$f(x) = \sum_{i=1}^{6} \alpha_i \overset{\phi(x_i)}{\underbrace{k(x_i, x)}}.$$

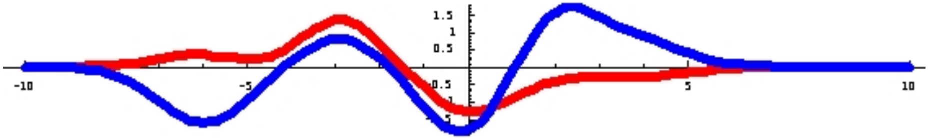


RBF Predictions

- f is a linear combination of 6 basis functions of form $k(x_i, \cdot)$:



- Predictions of the form $f(x) = \sum_{i=1}^{6} \alpha_i k(x_i, x)$:



- When kernelizing with RBF kernel, prediction functions always look this way (whether we get w from SVM, ridge regression).

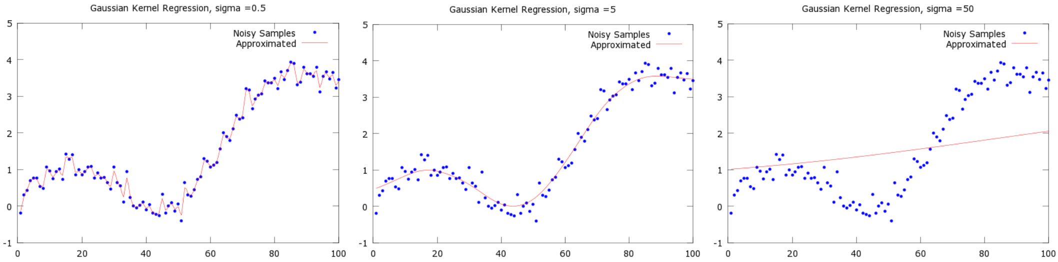
Effect of the bandwidth

How does the fitted function change when we vary the bandwidth parameter?

https://mccormickml.com/2014/02/26/kernel-regression/

Effect of the bandwidth

How does the fitted function change when we vary the bandwidth parameter?



(a) $\alpha = 0.5$ (b) $\alpha = 5$ (c) $\alpha = 50$

https://mccormickml.com/2014/02/26/kernel-regression/

Feature map of RBF kernel

What feature map corresponds to the RBF kernel?

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

Consider the 1D case ($x \in R$) where $\sigma = 1$:

$$k(x, x') = \exp\left(-\frac{(x - x')^2}{2}\right) \tag{1}$$

$$\phi(x)_i = e^{\frac{-x^2}{2}} \cdot \frac{x^i}{\sqrt{i}}$$

$$= \underbrace{\exp\left(-\frac{x^2}{2}\right)}_{f(x)} \underbrace{\exp\left(xx'\right)}_{\checkmark} \underbrace{\exp\left(-\frac{x'^2}{2}\right)}_{f(x')} \tag{2}$$

$$\exp(xx') = \sum_{i=0}^{\infty} \frac{(xx')^i}{i!}$$

$$g(x)_i = \frac{x^i}{\sqrt{i!}} \qquad \langle g(x), g(x') \rangle = \sum_{i=1}^{\infty} g(x)_i \, g(x')_i$$

$$= \exp(xx')$$

Based on https://www.cs.ubc.ca/~schmidtm/Courses/540-W19/L12.5.pdf

Kernel Methods

- A method can be kernelized if both training and inference only need inner produc in the feature space.
- Representer theorem says that all norm-regularized linear models can be kernelized.
    - Although we might be in a high dimensional space, w lies in the subspace spanned by $\phi(x_i)$.
    - Dimension of the subspace grows with the dataset size.
- Many other algorithms can be kernelized.

Kernelized perceptron

$w^* = \sum_i \alpha_i x_i$

Prediction:

$w \cdot x = \sum_i \alpha_i x_i \cdot x$

$= \sum_i \alpha_i \langle x_i, x \rangle$

$= k_x^T \alpha$

- Initialize $w \leftarrow 0$
- While not converged
  - For $(x_i, y_i) \in \mathcal{D}$
    - If $y_i w^T x_i < 0$ $\longrightarrow$ $y_i k_x^T \alpha < 0$
    - Update $w \leftarrow w + y_i x_i$ $\longrightarrow$ $\alpha_i \leftarrow \alpha_i + y_i$

$\sum_i \alpha_i x_i \leftarrow \sum_i \alpha_i x_i + y_i x_i$

Other kernel methods

- Distance-based methods depending on $\|x - x'\|^2$ $\langle x - x', x - x' \rangle$
  $= \langle x, x \rangle - 2\langle x, x' \rangle + \langle x', x' \rangle$
  - k-means clustering
  - k-nearest neighbors
- Eigenvalue methods: can show that eigenvector is in the span of data
  - Principal component analysis
  - Spectral clustering

Kernel SVM vs ridge

- For both kernel SVM and ridge regression, we make predictions by

$$\hat{f}(x) = k_x^T \alpha^* = \sum_{i=1}^{n} \alpha_i^* k(x_i, x)$$

- For SVM, we have sparsity in α^* from complementary slackness.
- For ridge, we need to access all training examples.
- For large-scale dataset, we may not be able to store/compute the kernel matrix.
    - Large-scale kernel machines (e.g. Random Features for Large-Scale Kernel Machines)