Chapter 4, Part 1: Informal Introduction to Kernels

Advanced Topics in Statistical Machine Learning

Tom Rainforth

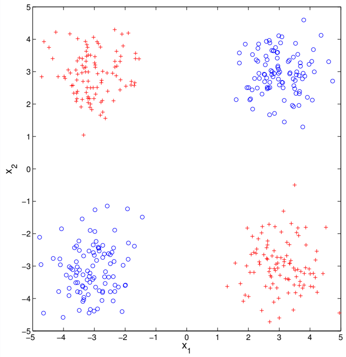
Hilary 2024

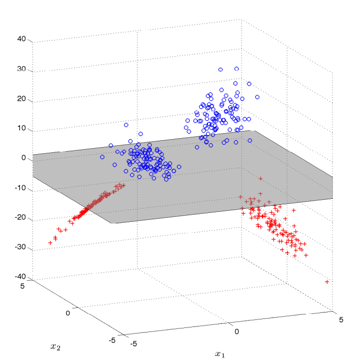
rainforth@stats.ox.ac.uk

Kernels Methods

- **Kernel methods** are a powerful class of **non–linear** machine learning algorithms
- They are based around employing **linear methods** in **nonlinearly** transformed **feature spaces**
    - If a linear method works with x directly, the corresponding kernel method will work with $\varphi(x)$ where $\varphi\colon \mathcal{X} \mapsto \mathcal{H}$ is a **feature map** to a higher dimensional feature space $\mathcal{H}$
- Typically we actually use a feature space $\mathcal{H}$ that is **infinite** dimensional!
- This is made possible by something known as the **kernel trick** that allows us to sidestep ever having to calculate $\varphi(x)$ directly

Beyond Linear Classifiers





- No linear classifier separates red from blue.
- Linear separation after mapping to a **higher dimensional feature space**:

$$x \;\mapsto\; \varphi(x) = \begin{pmatrix} x^{(1)} & x^{(2)} & x^{(1)}x^{(2)} \end{pmatrix}^\top \in \mathbb{R}^3$$

Revisiting SVMs

Dual program

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^\top x_j \quad \text{s.t.} \quad \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \preceq \alpha \preceq C \end{cases}$$

only depends on inputs x_i through their inner products $x_i^\top x_j$

Decision function also depends only on $x_i^\top x$

$$\hat{y}(x) = \text{sign}(w^\top x + b) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i x_i^\top x + b\right)$$

because our hyperplane definition is given by (using margin to denote the index of an arbitrary margin support vector)

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad b = y_{\text{margin}} - \sum_{i=1}^{n} \alpha_i y_i x_i^\top x_{\text{margin}}$$

SVMs with Nonlinear Features

Dual program

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \varphi(x_i)^\top \varphi(x_j) \quad \text{s.t.} \quad \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \preceq \alpha \preceq C \end{cases}$$

Decision function

$$\hat{y}(x) = \operatorname{sign}(w^\top \varphi(x) + b) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)^\top \varphi(x) + b \right)$$

where

$$w = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i), \quad b = y_{\text{margin}} - \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)^\top \varphi(x_{\text{margin}})$$

SVMs with Nonlinear Features

Dual program

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \varphi(x_i)^\top \varphi(x_j) \quad \text{s.t.} \quad \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \preceq \alpha \preceq C \end{cases}$$

Decision function

$$\hat{y}(x) = \operatorname{sign}(w^\top \varphi(x) + b) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)^\top \varphi(x) + b \right)$$

Key idea: for certain choices of φ, we can calculate $k(x, x') := \varphi(x)^\top \varphi(x')$ directly without ever needing to calculate $\varphi(x)$ or $\varphi(x')$. The corresponding k is known as a **kernel**.

Kernel SVMs

Dual program

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad \text{s.t} \quad \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \preceq \alpha \preceq C \end{cases}$$

Feature map $\varphi(x)$ still present, but implicit and never computed: we can use any k that implies a φ, even if the latter is unknown.

Decision function

$$\hat{y}(x) = \operatorname{sign}(w^\top \varphi(x) + b) = \operatorname{sign}\left(\sum_{i=1}^{n} \alpha_i y_i k(x, x_i) + b \right)$$

where we can further derive b using any of the support vectors:

$$b = y_{\text{margin}} - \sum_{i=1}^{n} \alpha_i y_i k(x_i, x_{\text{margin}}).$$

A Simple Example Kernel

- Suppose we have $p = 2$ dimensional problem, and we would like to introduce quadratic non-linearities,

$$\varphi(x) = \left(1, \sqrt{2}x^{(1)}, \sqrt{2}x^{(2)}, \sqrt{2}x^{(1)}x^{(2)}, \left(x^{(1)}\right)^2, \left(x^{(2)}\right)^2\right)^\top.$$

Then

$$\begin{aligned}\varphi(x_i)^\top\varphi(x_j) &= 1 + 2x_i^{(1)}x_j^{(1)} + 2x_i^{(2)}x_j^{(2)} + 2x_i^{(1)}x_i^{(2)}x_j^{(1)}x_j^{(2)} \\ &\quad + \left(x_i^{(1)}\right)^2\left(x_j^{(1)}\right)^2 + \left(x_i^{(2)}\right)^2\left(x_j^{(2)}\right)^2 \\ &= (1 + x_i^\top x_j)^2\end{aligned}$$

- We can thus calculate our inner products between features using the kernel $k(x_i, x_j) = \varphi(x_i)^\top\varphi(x_j) = (1 + x_i^\top x_j)^2$
- d-order interactions can be similarly be implemented by $k(x_i, x_j) = (1 + x_i^\top x_j)^d$ (**polynomial kernel**)

A More Powerful Example

- Consider the **infinite dimensional** mapping (assuming $p = 1$)

$$\varphi(x) = \exp\left(-\frac{1}{2}x^2\right)\left[1, x, \frac{x^2}{\sqrt{2!}}, \frac{x^3}{\sqrt{3!}}, \ldots, \frac{x^r}{\sqrt{r!}}, \ldots\right]^\top$$

- Here we have

$$\begin{aligned}\varphi(x_i)^\top\varphi(x_j) &= \exp\left(-\frac{1}{2}x_i^2\right)\exp\left(-\frac{1}{2}x_j^2\right)\sum_{r=0}^{\infty}\frac{x_i^r x_j^r}{r!} \\ &= \exp\left(-\frac{1}{2}x_i^2\right)\exp\left(-\frac{1}{2}x_j^2\right)\exp(x_i x_j) \\ &= \exp\left(-\frac{1}{2}\left(x_i - x_j\right)^2\right) =: k(x_i, x_j).\end{aligned}$$

- This is known as the exponential (or RBF) kernel. Its general form that allows for more dimensions and a scaling term γ:

$$k(x_i, x_j) = \exp\left(-\frac{1}{2\gamma^2}\|x_i - x_j\|_2^2\right)$$

The Kernel Trick

The **kernel trick** states that if we have an algorithm that relies only on inner products of our data $x_i^\top x_j$, then we can apply this algorithm on a (potentially infinite) nonlinear feature mapping of our data by replacing all such calculations with a kernel $k(x_i, x_j)$.

This kernel must imply some feature map $\varphi(x)$ such that $k(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle$, where $\langle \cdot, \cdot \rangle$ denotes an inner product.[1] However, we need not be able to calculate $\varphi(x_i)$, or indeed know the definition of φ.

Lots of base algorithms turn out to be amenable to this approach: e.g. SVMs, linear regression, logistic regression, PCA, CCA, ICA, K-means.

[1] For example, $\langle \varphi(x_i), \varphi(x_j) \rangle = \varphi(x_i)^\top \varphi(x_j)$. See next lecture for a more precise definition.

[Examples demo]