Formalize Kernel Trick

Consider the same example: $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$, and define $\phi(\mathbf{x}) = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_d \\ x_1 x_2 \\ \vdots \\ x_{d-1} x_d \\ x_1 x_2 x_3 \\ \vdots \\ x_1 x_2 \cdots x_d \end{pmatrix}$.

The inner product $\phi(\mathbf{x})^\top \phi(\mathbf{z})$ can be formulated as:

$$\phi(\mathbf{x})^\top \phi(\mathbf{z}) = 1 \cdot 1 + x_1 z_1 + x_2 z_2 + \cdots + x_1 x_2 z_1 z_2 + \cdots + x_1 \cdots x_d z_1 \cdots z_d = \prod_{j=1}^{d} (1 + x_j z_j$$

This is a remarkable achievement: the explicit sum of 2^d terms becomes the product of d terms. Hence, we can compute the inner-product from the above formula in the order of d operations instead of the order of 2^d.

We call this abstract inner product function the kernel function:

$$\underbrace{\mathsf{K}(\mathbf{x}, \mathbf{z})}_{\text{kernel function}} = \phi(\mathbf{x})^\top \phi(\mathbf{z})$$

With a finite training set of n samples, inner products are often pre-computed and stored in a Kernel Matrix with the ij^{th} entry $\mathsf{K}_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$.