Kernel methods

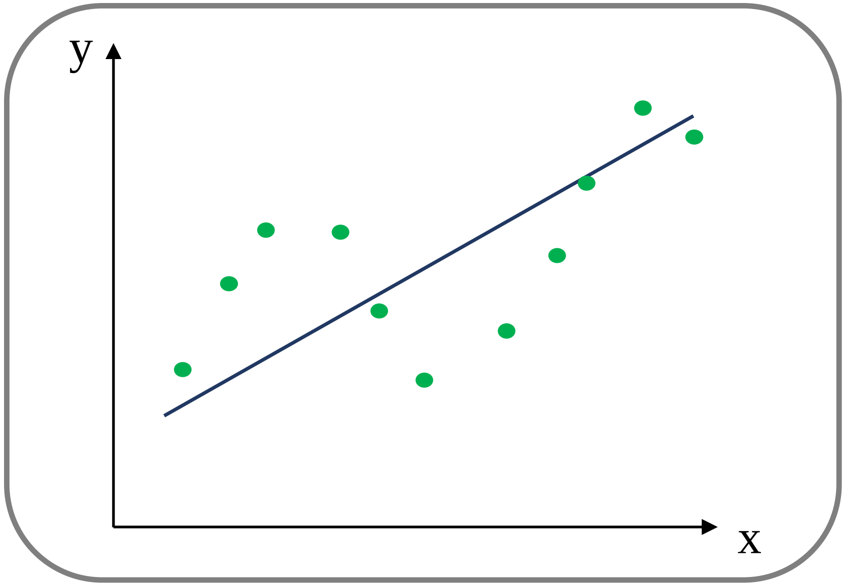
Dripta Mj
Department of Mathematics
Ramakrishna Mission Vivekananda Educational and Research Institute
Belur Math, India

Machine Learning
Sem 3, 2018-19

Sem 3, 2018-19

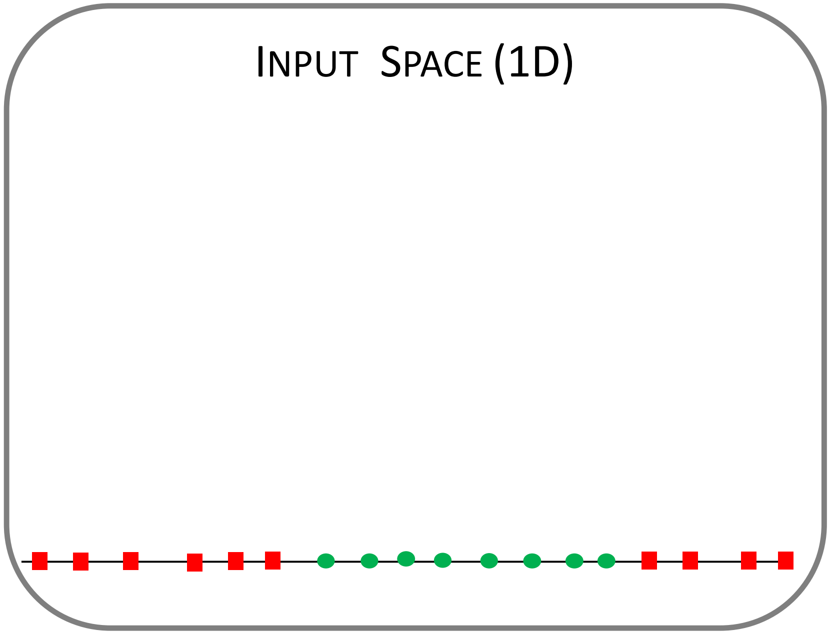
Introduction

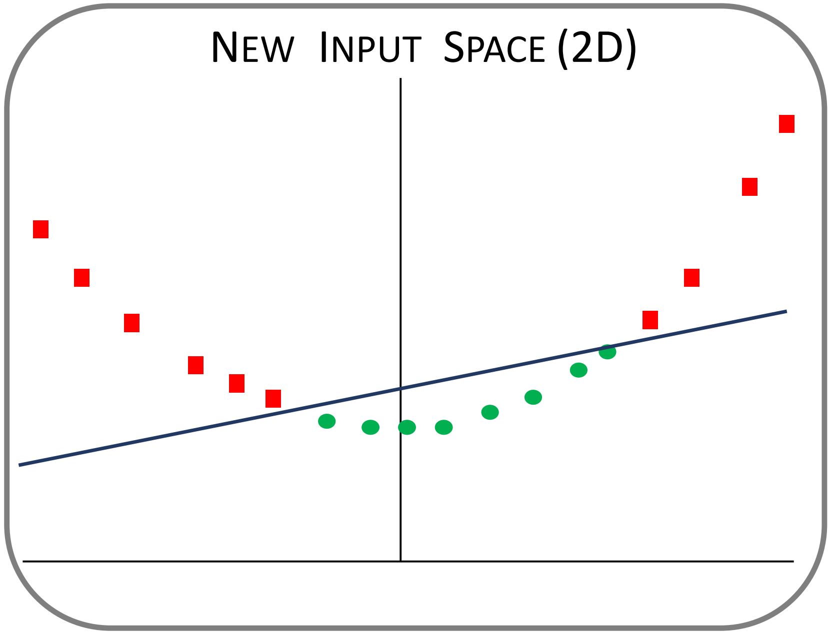
- Structures in real-world data are often non-linear.
  - Linear models are not suitable in such cases.



- Kernels project data to a higher dimensional space where the structures are linear.
  - The transformation facilitates application of linear models in the new space.
- Explicit evaluation of feature mappings can be computationally expensive, but kernel methods overcome the issue....

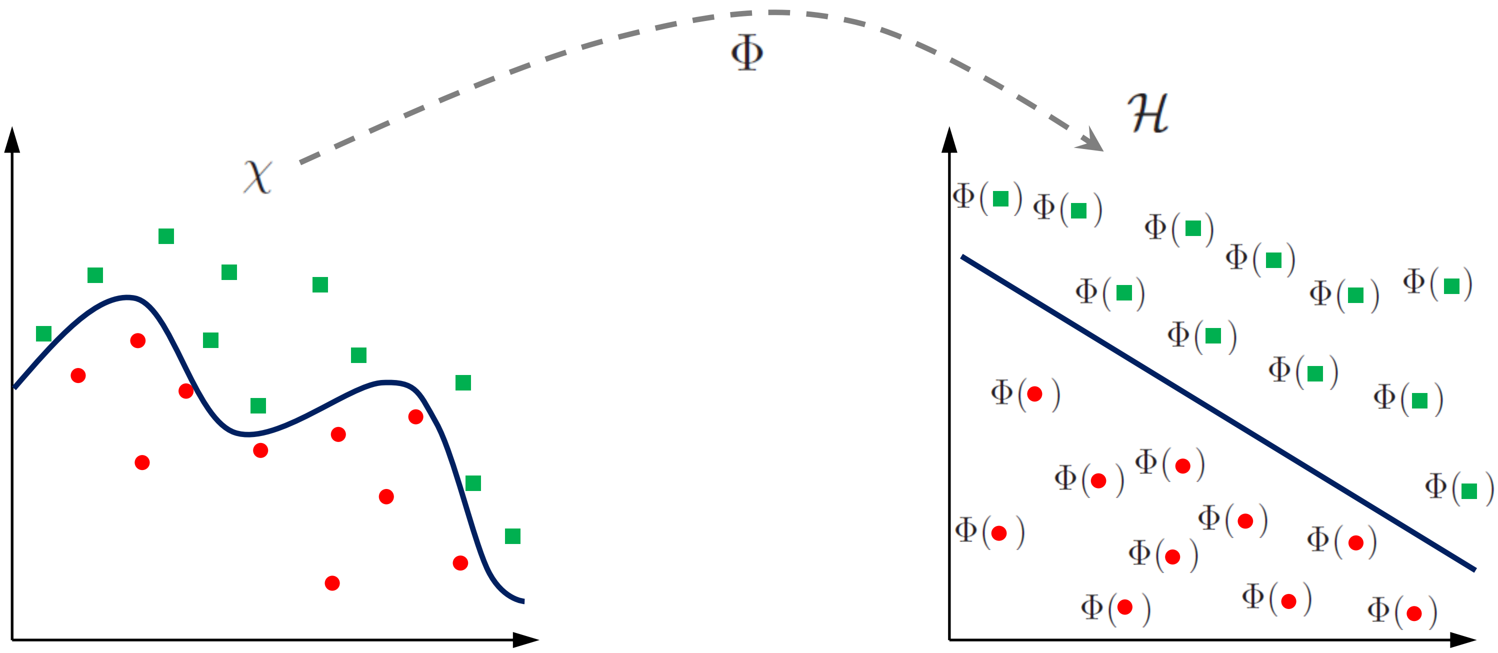
Binary classification problem



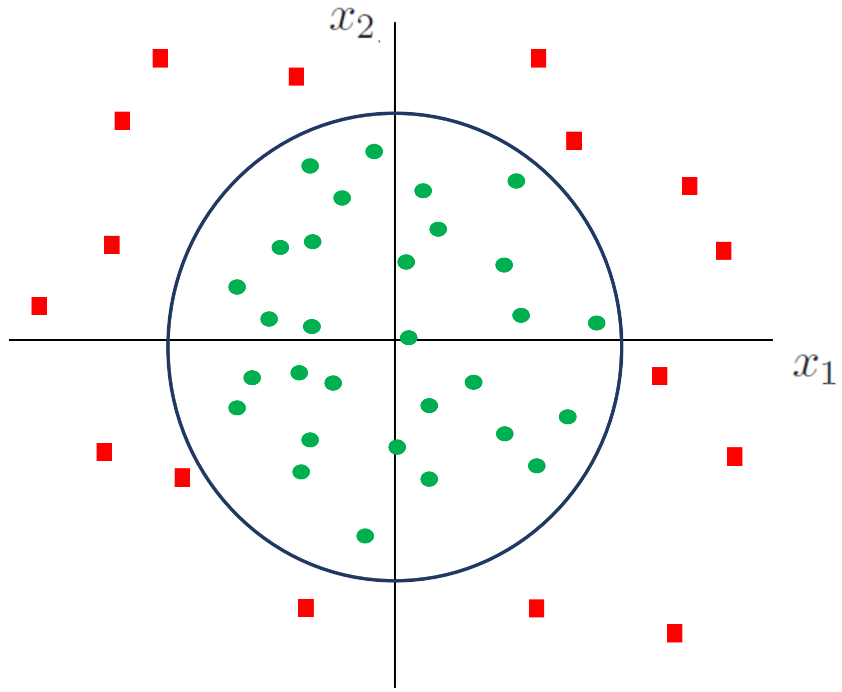


- Linear separation of data is not possible.
- Consider the following mapping: $\Phi(x): \; x \rightarrow [x, \; x^2]$
- The dimension of the new input space is 2 as there are two features.
- Data linearly separable in the new input space.

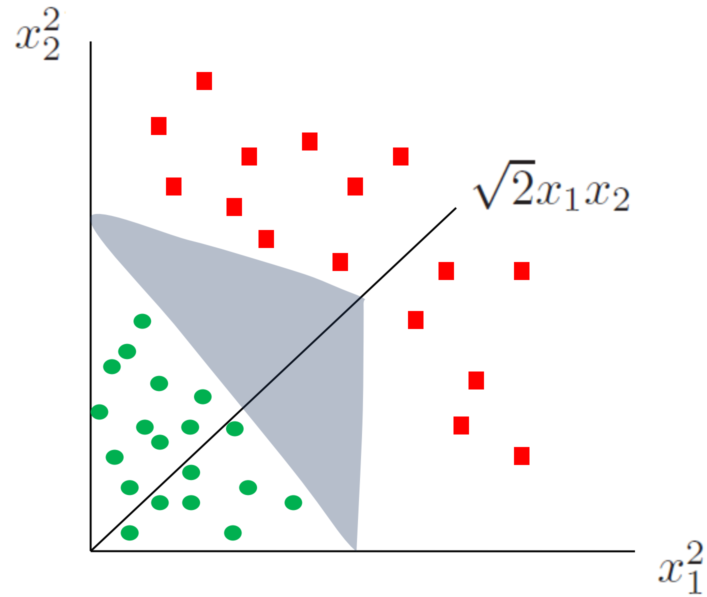
Mapping



Example



- Input space: $\mathbf{x} = [x_1 \;\; x_2]$.
- Data **not** linearly separable in input space.



- Feature space: $\Phi(\mathbf{x}) = [x_1^2 \;\; \sqrt{2}x_1x_2 \;\; x_2^2]$.
- Data linearly separable in feature space.

Kernels

- From the previous example we have

$$\Phi(\mathbf{x}^{(i)}) = \left[(\mathbf{x}_1^{(i)})^2 \quad \sqrt{2}\mathbf{x}_1^{(i)}\mathbf{x}_2^{(i)} \quad (\mathbf{x}_2^{(i)})^2 \right] \quad \text{and} \quad \Phi(\mathbf{x}^{(j)}) = \left[(\mathbf{x}_1^{(j)})^2 \quad \sqrt{2}\mathbf{x}_1^{(j)}\mathbf{x}_2^{(j)} \quad (\mathbf{x}_2^{(j)})^2 \right]$$

- The inner product of $\Phi\left(\mathbf{x}^{(i)}\right)$ and $\Phi\left(\mathbf{x}^{(j)}\right)$ yields

$$\begin{aligned}
\left\langle \Phi\left(\mathbf{x}^{(i)}\right), \Phi\left(\mathbf{x}^{(j)}\right) \right\rangle &= \left\langle \left[(\mathbf{x}_1^{(i)})^2 \quad \sqrt{2}\mathbf{x}_1^{(i)}\mathbf{x}_2^{(i)} \quad (\mathbf{x}_2^{(i)})^2 \right], \left[(\mathbf{x}_1^{(j)})^2 \quad \sqrt{2}\mathbf{x}_1^{(j)}\mathbf{x}_2^{(j)} \quad (\mathbf{x}_2^{(j)})^2 \right] \right\rangle \\
&= (\mathbf{x}_1^{(i)})^2 (\mathbf{x}_1^{(j)})^2 + 2\mathbf{x}_1^{(i)}\mathbf{x}_2^{(i)}\mathbf{x}_1^{(j)}\mathbf{x}_2^{(j)} + (\mathbf{x}_2^{(i)})^2 (\mathbf{x}_2^{(j)})^2 \\
&= \left(\mathbf{x}_1^{(i)}\mathbf{x}_1^{(j)} + \mathbf{x}_2^{(i)}\mathbf{x}_2^{(j)} \right)^2 \\
&= < \mathbf{x}^{(i)}, \mathbf{x}^{(j)} >^2
\end{aligned}$$

- So the kernel function is

$$K\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = < \mathbf{x}^{(i)}, \mathbf{x}^{(j)} >^2$$

Kernels

- High dimensional mapping can lead to large number of features.
  - Computing the mapping and using the mapped representation could be inefficient.

- Kernels address these shortcomings.

- Kernels implicitly define a mapping to a high dimensional space

$$K(\mathbf{x}, \mathbf{x}') =< \Phi(\mathbf{x}), \Phi(\mathbf{x}') >.$$

- Kernel $K(\mathbf{x}, \mathbf{x}')$ efficiently computes the inner product $< \Phi(\mathbf{x}), \Phi(\mathbf{x}') >$.

- Explicitly computing $\Phi(\mathbf{x}), \Phi(\mathbf{x}')$ and then doing the inner product is more expensive.

Reproducing Kernel Hilbert space

- The function $K : \chi \times \chi \mapsto \mathcal{R}$ is called a reproducing kernel of a Hilbert space $\mathcal{H}$ when the following two conditions are satisfied:
  - $\forall \mathbf{x} \in \chi$

$$K(\mathbf{x},\ ) = K_{\mathbf{x}} \in \mathcal{H}$$

    where $K_{\mathbf{x}}$ is a function of single variable with $\mathbf{x}$ fixed.

  - $\forall \mathbf{x} \in \chi$ and $\forall f \in \mathcal{H}$

$$< f, K_{\mathbf{x}} >_{\mathcal{H}} = f(\mathbf{x})$$

    This is called the **reproducing property**: Inner product of the functions f and $K_{\mathbf{x}}$ yields the evaluation of the function at the point $\mathbf{x}$.

- If such a kernel K exists then $\mathcal{H}$ is called a Reproducing Kernel Hilbert space (RKHS).

Reproducing kernel

- Theorem: Function $K : \chi \times \chi \mapsto \mathcal{R}$ is **positive definite** iff it is a **reproducing kernel**.
  - For $\mathbf{x} \in \chi$ and $\mathbf{x}' \in \chi$ we have:

$$K(\mathbf{x}, \mathbf{x}') =< K_{\mathbf{x}}, K_{\mathbf{x}'} >_{\mathcal{H}}$$
$$=< K_{\mathbf{x}'}, K_{\mathbf{x}} >_{\mathcal{H}} = K(\mathbf{x}', \mathbf{x})$$

  - For $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N) \in \chi^N$, and $(a_1, a_2, ..., a_N) \in \mathcal{R}^N$ we have

$$\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j < K_{\mathbf{x}_i}, K_{\mathbf{x}_j} >_{\mathcal{H}}$$
$$= ||\sum_{i=1}^{N} a_i K_{\mathbf{x}_i}||^2_{\mathcal{H}}$$
$$\geq 0$$

Reproducing kernel

- Theorem (Aronszajn): For a positive definite kernel K on set χ there exists a Hilbert space $\mathcal{H}$ and a mapping

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

such that

$$K(\mathbf{x}, \mathbf{x}') =< \Phi(\mathbf{x}), \Phi(\mathbf{x}') >_{\mathcal{H}}$$

for $\forall \mathbf{x} \in \chi$ and $\mathbf{x}' \in \chi$.

  - Consider the mapping $\Phi : \chi \mapsto \mathcal{H}$ such that $\forall \mathbf{x} \in \chi$: $\Phi(\mathbf{x}) = K_{\mathbf{x}}$.
  - Then the reproducing property yields:

$$\begin{aligned} < \Phi(\mathbf{x}), \Phi(\mathbf{x}') >_{\mathcal{H}} &= < K_{\mathbf{x}}, K_{\mathbf{x}'} >_{\mathcal{H}} \\ &= K(\mathbf{x}, \mathbf{x}') \end{aligned}$$

Kernel combinations

- If K_1 and K_2 are positive definite kernels, then the following combinations
  - $K(\mathbf{x}, \mathbf{x}') = K_1(\mathbf{x}, \mathbf{x}') + K_2(\mathbf{x}, \mathbf{x}')$,
  - $K(\mathbf{x}, \mathbf{x}') = K_1(\mathbf{x}, \mathbf{x}')K_2(\mathbf{x}, \mathbf{x}')$,
  - $K(\mathbf{x}, \mathbf{x}') = \beta K_1(\mathbf{x}, \mathbf{x}')$, where $\beta \geq 0$.
- New kernels can be created by using the above rules.

Sum of kernels

$$\begin{aligned} k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') &= < \Phi_1(\mathbf{x}), \Phi_1(\mathbf{x}') > + < \Phi_2(\mathbf{x}), \Phi_2(\mathbf{x}') > \\ &= < [\Phi_1(\mathbf{x})\Phi_2(\mathbf{x})], [\Phi_1(\mathbf{x}')\Phi_2(\mathbf{x}')] > \\ &= k_3(\mathbf{x}, \mathbf{x}') \end{aligned}$$

- The summation of the two kernels corresponds to the concatenation of their respective feature spaces.

Product of kernels

$$k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') = \sum_{p=1}^{P} \Phi_{1p}(\mathbf{x})\Phi_{1p}(\mathbf{x}') \sum_{m=1}^{M} \Phi_{2m}(\mathbf{x})\Phi_{2m}(\mathbf{x}')$$

$$= \sum_{p=1}^{P} \sum_{m=1}^{M} \Big(\Phi_{1p}(\mathbf{x})\Phi_{2m}(\mathbf{x})\Big)\Big(\Phi_{1p}(\mathbf{x}')\Phi_{2m}(\mathbf{x}')\Big)$$

$$= \sum_{k=1}^{PM} \Big(\Phi_{12k}(\mathbf{x})\Phi_{12k}(\mathbf{x}')\Big)$$

where $\Phi_{12}(\mathbf{x}) = \Phi_1(\mathbf{x})\Phi_2(\mathbf{x})$ is the Cartesian product

$$= < \Phi_{12}(\mathbf{x}), \Phi_{12}(\mathbf{x}') >$$

$$= k_3(\mathbf{x}, \mathbf{x}')$$

Gaussian kernel

$$
\begin{aligned}
k(\mathbf{x},\mathbf{x}') &= \exp\left[-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2l^2}\right] \\
&= \exp\left[-\frac{<\mathbf{x}-\mathbf{x}',\mathbf{x}-\mathbf{x}'>}{2l^2}\right] \\
&= \exp\left[-\frac{<\mathbf{x},\mathbf{x}-\mathbf{x}'> - <\mathbf{x}',\mathbf{x}-\mathbf{x}'>}{2l^2}\right] \\
&= \exp\left[-\frac{<\mathbf{x},\mathbf{x}> - <\mathbf{x},\mathbf{x}'> - <\mathbf{x}',\mathbf{x}> + <\mathbf{x}',\mathbf{x}'>}{2l^2}\right] \\
&= \exp\left[-\frac{||\mathbf{x}||^2 + ||\mathbf{x}'||^2 - 2<\mathbf{x},\mathbf{x}'>}{2l^2}\right] \\
&= \left(\exp\left[-\frac{||\mathbf{x}||^2}{2l^2}\right]\exp\left[-\frac{||\mathbf{x}'||^2}{2l^2}\right]\right)\exp\left[\frac{<\mathbf{x},\mathbf{x}'>}{l^2}\right] \\
&= k_1(\mathbf{x},\mathbf{x}')\left[1+\frac{<\mathbf{x},\mathbf{x}'>}{l^2}+\frac{<\mathbf{x},\mathbf{x}'>^2}{2l^4}+\frac{<\mathbf{x},\mathbf{x}'>^3}{3l^6}+\ldots\ldots\right] \\
&= k_1(\mathbf{x},\mathbf{x}')\sum_{n=0}^{\infty}\frac{<\mathbf{x},\mathbf{x}'>^n}{n!} \qquad = k_1(\mathbf{x},\mathbf{x}')\sum_{n=0}^{\infty}\frac{k_{\mathrm{poly}(n)}(\mathbf{x},\mathbf{x}')}{n!}
\end{aligned}
$$

Kernel trick

- Algorithms that can be expressed in terms of pairwise inner products of inputs can also be applied to the feature space of a kernel by replacing the inner product with a kernel evaluation.
- This is possible because the kernel is an inner product in the feature space.

Representer theorem (simplified)

- Given a set χ, a kernel k, corresponding RKHS $\mathcal{H}$, and a (loss) function $\mathcal{L}(. \; , \; .)$, the solutions of the optimization problem

$$\arg\min_{f\in\mathcal{H}} \sum_{n=1}^{N} \mathcal{L}\left(f\left(\mathbf{x}^{(n)}\right), y^{(n)}\right) + \lambda||f||_{\mathcal{H}}^{2}$$

admits the following representation

$$f = \sum_{n=1}^{N} \alpha_n k\left(\mathbf{x}^{(n)}, \; . \; \right)$$

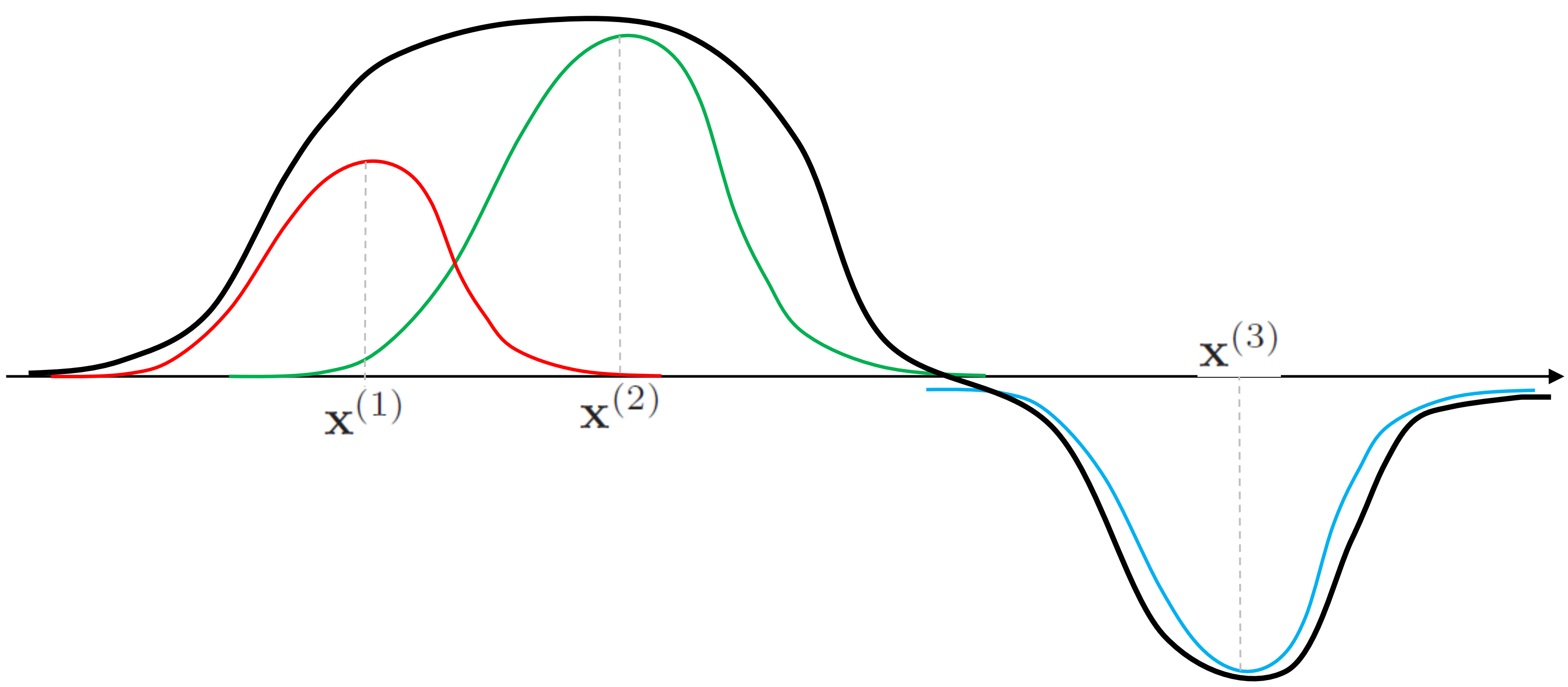
$$\Rightarrow f(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n k\left(\mathbf{x}^{(n)}, \mathbf{x}\right)$$

Although the optimization problem can potentially be in an infinite dimensional space $\mathcal{H}$, the solution lies in the span of N kernels centered at the N data points.

Representer theorem

If we are given three input points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ and $\mathbf{x}^{(3)}$, then we have

$$f(\mathbf{x}) = \alpha_1 k(\mathbf{x}^{(1)}, \mathbf{x}) + \alpha_2 k(\mathbf{x}^{(2)}, \mathbf{x}) + \alpha_3 k(\mathbf{x}^{(3)}, \mathbf{x})$$



Kernel-SVM

- Recall the SVM objective (dual formulation):

$$\max_{0 \leq \boldsymbol{\lambda} \leq C} -\frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \lambda_m \lambda_n y^{(m)} y^{(n)} ((\mathbf{x}^{(m)})^{\mathrm{T}} \mathbf{x}^{(n)}) + \sum_{n=1}^{N} \lambda_n \quad \text{subject to} \quad \sum_{n=1}^{N} \lambda_n y^{(n)} = 0$$

- Let Φ be the feature map corresponding to some kernel k. Then

$$k(\mathbf{x}^{(m)}, \mathbf{x}^{(n)}) = < \Phi(\mathbf{x}^{(m)}), \Phi(\mathbf{x}^{(n)}) >$$

- Replacing the inner product $< \mathbf{x}^{(m)}, \mathbf{x}^{(n)} >$ in the objective function by $< \Phi(\mathbf{x}^{(m)}), \Phi(\mathbf{x}^{(n)}) >$ yields

$$\max_{0 \leq \boldsymbol{\lambda} \leq C} -\frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \lambda_m \lambda_n y^{(m)} y^{(n)} < \Phi(\mathbf{x}^{(m)}), \Phi(\mathbf{x}^{(n)}) > + \sum_{n=1}^{N} \lambda_n$$

$$\text{subject to} \quad \sum_{n=1}^{N} \lambda_n y^{(n)} = 0$$

- By using this formulation the SVM learns a linear separator in the feature space $\mathcal{H}$ which corresponds to a non-linear decision boundary in the original input space.

Kernel-SVM

- Solution to $\mathbf{w}$ in original SVM formulation:

$$\mathbf{w} = \sum_{n=1}^{N} \lambda_n y^{(n)} \mathbf{x}^{(n)}$$

- Solution to $\mathbf{w}$ in kernel-SVM formulation:

$$\mathbf{w} = \sum_{n=1}^{N} \lambda_n y^{(n)} \Phi\big(\mathbf{x}^{(n)}\big) = \sum_{n=1}^{N} \lambda_n y^{(n)} k\big(\mathbf{x}^{(n)},\ .\ \big)$$

- Test prediction at $\mathbf{x}^*$ in original SVM formulation:

$$y^* = \mathrm{sign}\big(\mathbf{w}^{\mathrm{T}}\mathbf{x}\big) = \mathrm{sign}\Big(\sum_{n=1}^{N} \lambda_n y^{(n)} \big(\mathbf{x}^{(n)}\big)^{\mathrm{T}} \mathbf{x}\Big)$$

- Test prediction at $\mathbf{x}^*$ in kernel-SVM formulation:

$$\begin{aligned} y^* &= \mathrm{sign}\big(\mathbf{w}^{\mathrm{T}}\mathbf{x}\big) = \mathrm{sign}\Big(\sum_{n=1}^{N} \lambda_n y^{(n)} < \Phi\big(\mathbf{x}^{(n)}\big), \Phi\big(\mathbf{x}^*\big) > \Big) \\ &= \mathrm{sign}\Big(\sum_{n=1}^{N} \lambda_n y^{(n)} k\big(\mathbf{x}^{(n)}\big), \mathbf{x}^*\Big) \end{aligned}$$

Kernel matrix

- Also known as Gram matrix.

- Formed by applying the kernel function k to all pairs of data points in $\mathbf{X}$.

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}) & k(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) & \cdot & \cdot & k(\mathbf{x}^{(1)}, \mathbf{x}^{(N)}) \\ k(\mathbf{x}^{(2)}, \mathbf{x}^{(1)}) & k(\mathbf{x}^{(2)}, \mathbf{x}^{(2)}) & \cdot & \cdot & k(\mathbf{x}^{(2)}, \mathbf{x}^{(N)}) \\ k(\mathbf{x}^{(3)}, \mathbf{x}^{(1)}) & k(\mathbf{x}^{(3)}, \mathbf{x}^{(2)}) & \cdot & \cdot & k(\mathbf{x}^{(3)}, \mathbf{x}^{(N)}) \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ k(\mathbf{x}^{(N)}, \mathbf{x}^{(1)}) & k(\mathbf{x}^{(N)}, \mathbf{x}^{(2)}) & \cdot & \cdot & k(\mathbf{x}^{(N)}, \mathbf{x}^{(N)}) \end{bmatrix}$$

- Square matrix of size $N \times N$.

- Symmetric.