CIS 519 Recitation 5

Perceptron Updating

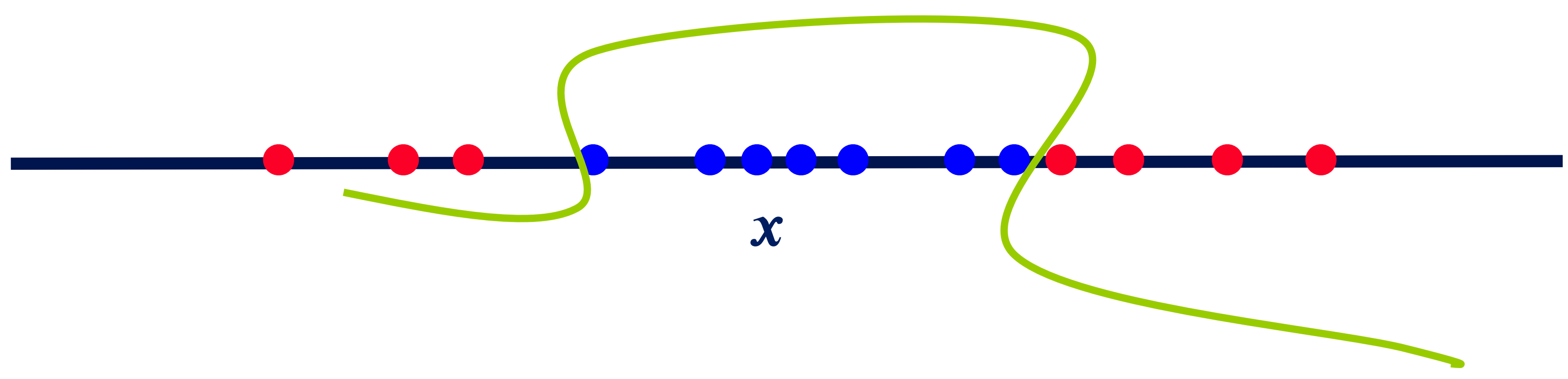
Algorithm **Perceptron**

Initial weight vector: $\mathbf{w}_1 = \mathbf{0} \in \mathbb{R}^d$
For $t = 1, \ldots, T$:
- Receive instance $\mathbf{x}_t \in \mathcal{X} \subseteq \mathbb{R}^d$
- Predict $\widehat{y}_t = \mathrm{sign}(\mathbf{w}_t^\top \mathbf{x}_t)$
- Receive true label $y_t \in \{\pm 1\}$
- Incur loss $\mathbf{1}(\widehat{y}_t \neq y_t)$
- Update: If $\widehat{y}_t \neq y_t$ then
  $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \, \mathbf{x}_t$
  else
  $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$

- Assume you made a mistake on example $\boldsymbol{x}$. => $y_t(\mathbf{w}_t^T\mathbf{x}_t) < 0$
- $y_t(\mathbf{w}_{t+1}^T\mathbf{x}_t) = y_t[(\mathbf{w}_t + y_t\mathbf{x}_t)^T\mathbf{x}_t] = y_t\mathbf{w}_t^T\mathbf{x}_t + y_t^2\mathbf{x}_t^T\mathbf{x}_t > y_t\mathbf{w}_t^T\mathbf{x}_t$
- You then see example $\boldsymbol{x}$ again; will you make a mistake on it?

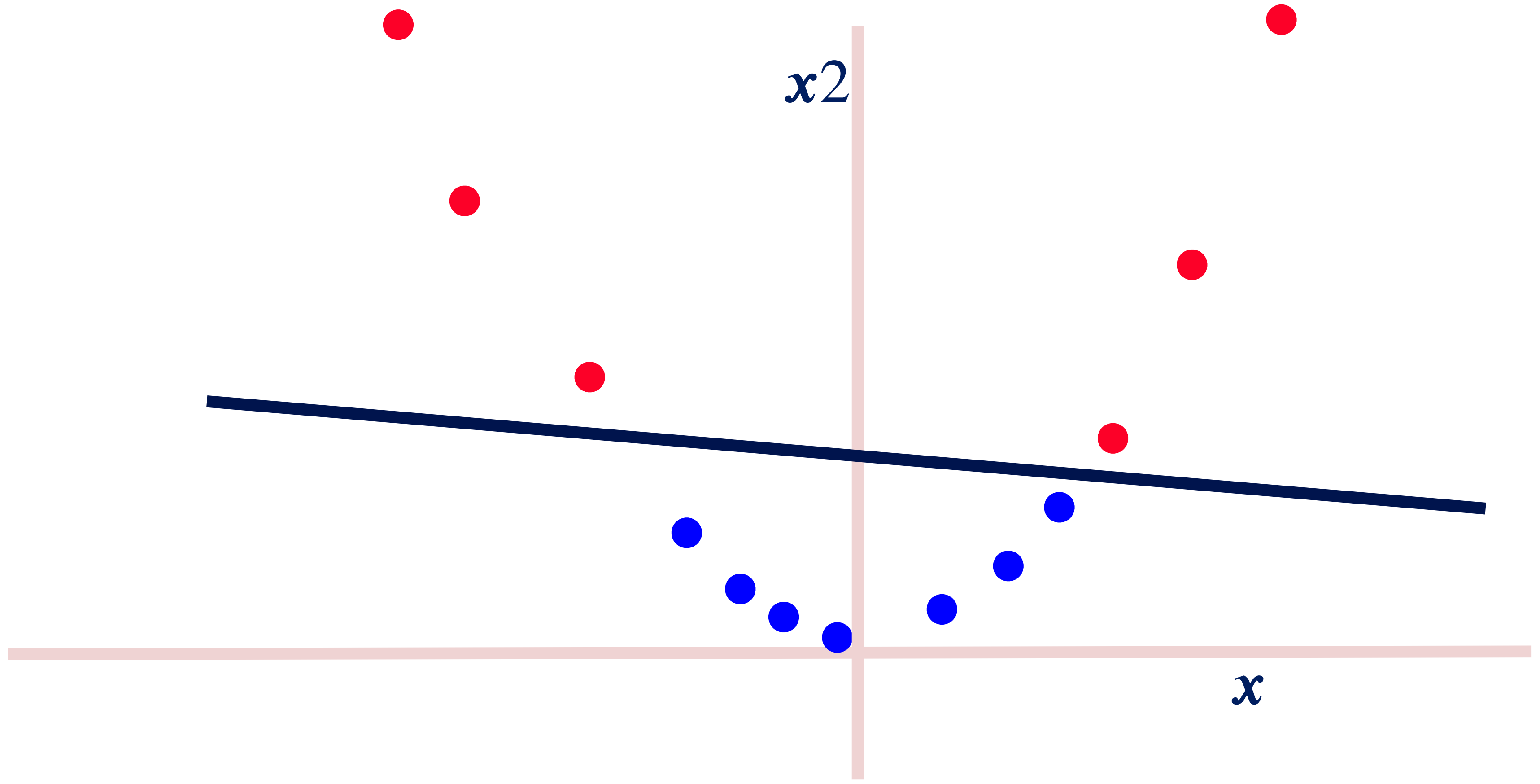
Why do we need kernel?

- Data are not linearly separable in one dimension
- Not separable if you insist on using a specific class of functions



Why do we need kernel?

- Data are separable in space



Why do we need the kernel trick?

- Prediction with respect to a separating hyper planes (produced by Perceptron, SVM) can be computed as a function of dot products of feature based representation of examples.
- We want to define a dot product in a high dimensional space.
- Given two examples $x = (x_1, x_2, \ldots x_n)$ and $y = (y_1, y_2, \ldots y_n)$ we want to map them to a high dimensional space [example- quadratic]:

$$\Phi(x_1, x_2, \ldots, x_n) = (1, \sqrt{2}x_1, \ldots, \sqrt{2}x_n, x_1^2, \ldots, x_n^2, \sqrt{2}x_1x_2, \ldots, \sqrt{2}x_{n-1}x_n)$$
$$\Phi(y_1, y_2, \ldots, y_n) = (1, \sqrt{2}y_1, \ldots, \sqrt{2}y_n, y_1^2, \ldots, y_n^2, \sqrt{2}y_1y_2, \ldots, \sqrt{2}y_{n-1}y_n)$$

and compute the dot product $A = \Phi(x)^T\Phi(y)$ [takes $O(n^2)$ time]

- Instead, in the original space, compute
- $B = k(x, y) = [1 + (x_1, x_2, \ldots x_n)^T (y_1, y_2, \ldots y_n)]^2$ [takes $O(n)$ time]
- Theorem: A = B

Kernel Examples: Questions

Let $K_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $K_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be two symmetric, positive definite kernel functions, and for simplicity, assume that each implements dot products in some finite-dimensional space, so that there are vector mappings $\phi_1 : \mathcal{X} \to \mathbb{R}^{d_1}$ and $\phi_2 : \mathcal{X} \to \mathbb{R}^{d_2}$ for some $d_1, d_2 \in \mathbb{Z}_+$ such that

$$K_1(x, x') = \phi_1(x)^\top \phi_1(x') , \qquad K_2(x, x') = \phi_2(x)^\top \phi_2(x') \quad \forall x, x' \in \mathcal{X} .$$

For each of the following functions $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, either find a vector mapping $\phi : \mathcal{X} \to \mathbb{R}^d$ for some suitable $d \in \mathbb{Z}_+$ such that $K(x, x') = \phi(x)^\top \phi(x')$ $\forall x, x'$, or explain why such a mapping cannot exist.

1. $K(x, x') = c \cdot K_1(x, x')$, where $c > 0$
2. $K(x, x') = K_1(x, x') + K_2(x, x')$
3. $K(x, x') = K_1(x, x') - K_2(x, x')$
4. $K(x, x') = K_1(f(x), f(x'))$, where $f : \mathcal{X} \to \mathcal{X}$ is any function.

Kernel Examples: Solutions

1. $\phi(x) = \sqrt{c} \cdot K_1(x, x')$
2. $\phi(x) = \begin{pmatrix} \phi_1(x) \\ \phi_2(x) \end{pmatrix}$
3. The difference of two positive definite matrices need not be a positive definite matrix, therefore in this case this is not a valid kernel, i.e. there does not in general exist a mapping ϕ satisfying the desired property.
4. $\phi(x) = \phi_1(f(x))$

Quiz3-Q4

- You are tasked with learning a new function over 10 Boolean variables; you believe that this function evaluates to True if and only if a subset of these variables (you don't know which, and how many) is 1. Your friend says that they have a good learning algorithm that can learn linear threshold units and suggest that you use it. Is this a good choice?
  - Yes, since the class of LTUs over 10 variables can express all the functions you care about
  - No, since the class of LTUs over 10 variables cannot express all the functions you care about. You should use Decision Trees
  - Yes, since all Boolean functions can be represented as LTUs.
  - No, since only neural networks can express the type of functions you care about