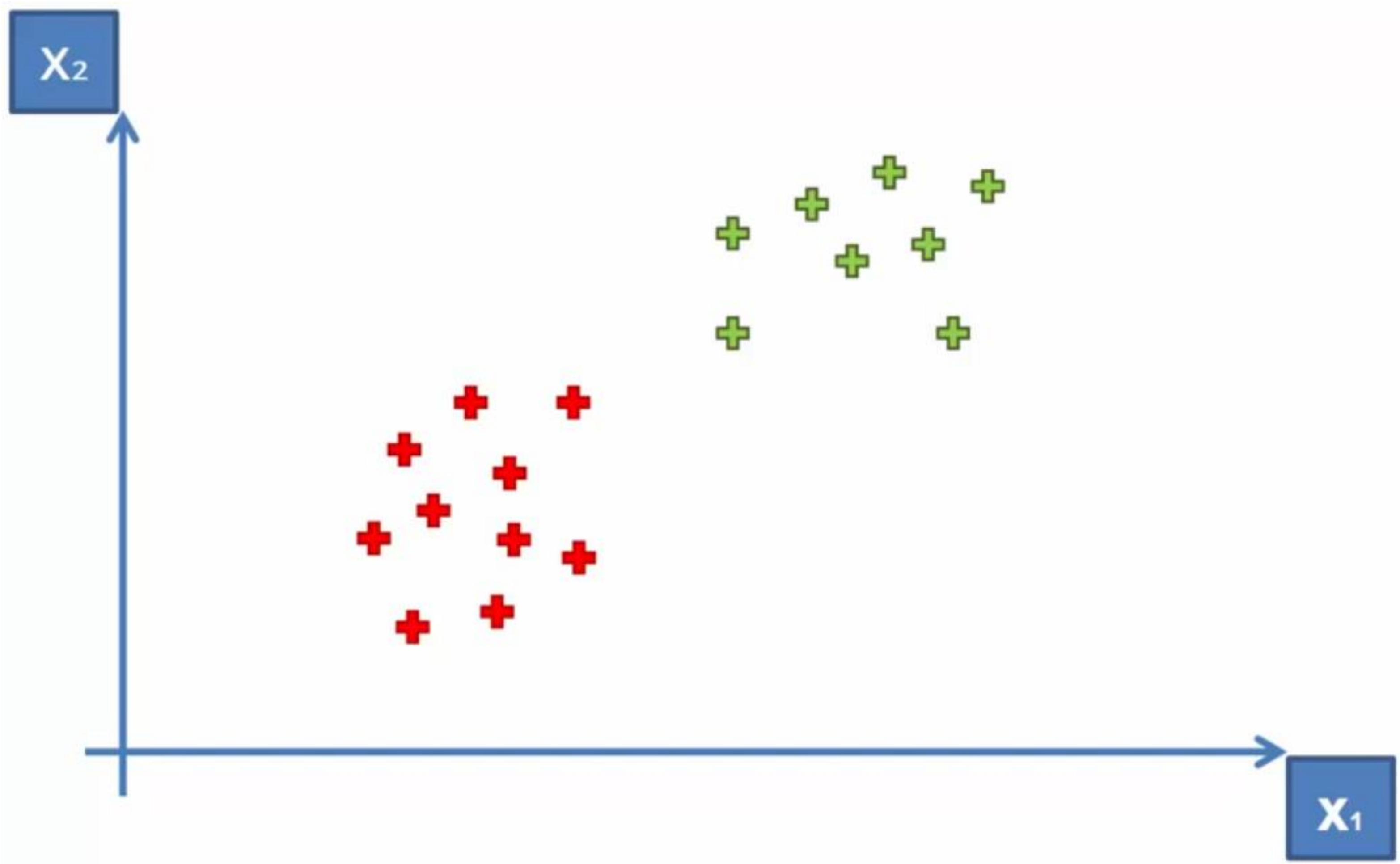
Introduction to Machine Learning. Lec.10 Support Vector Machines

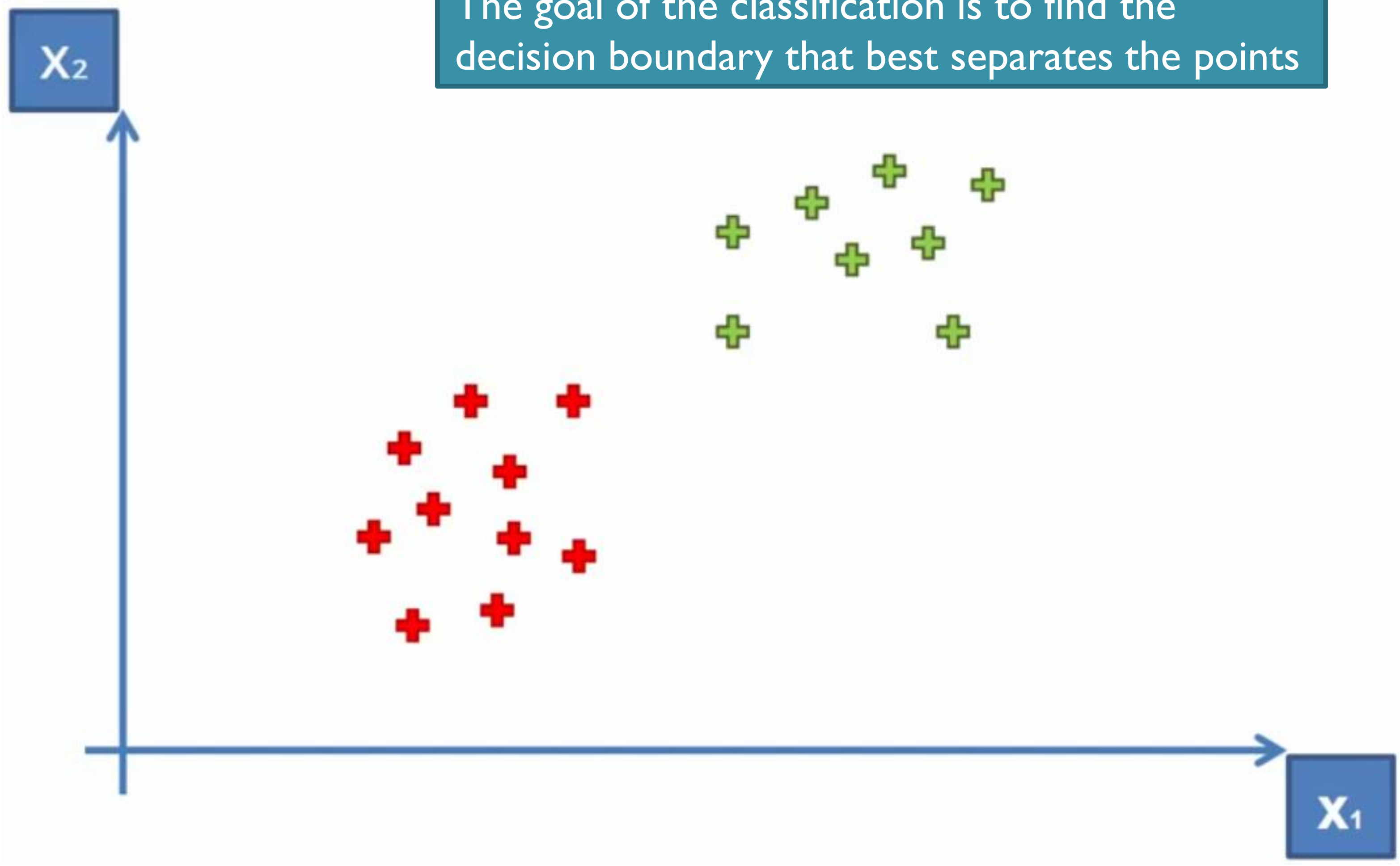
Aidos Sarsembayev, IITU, 2018

Classification



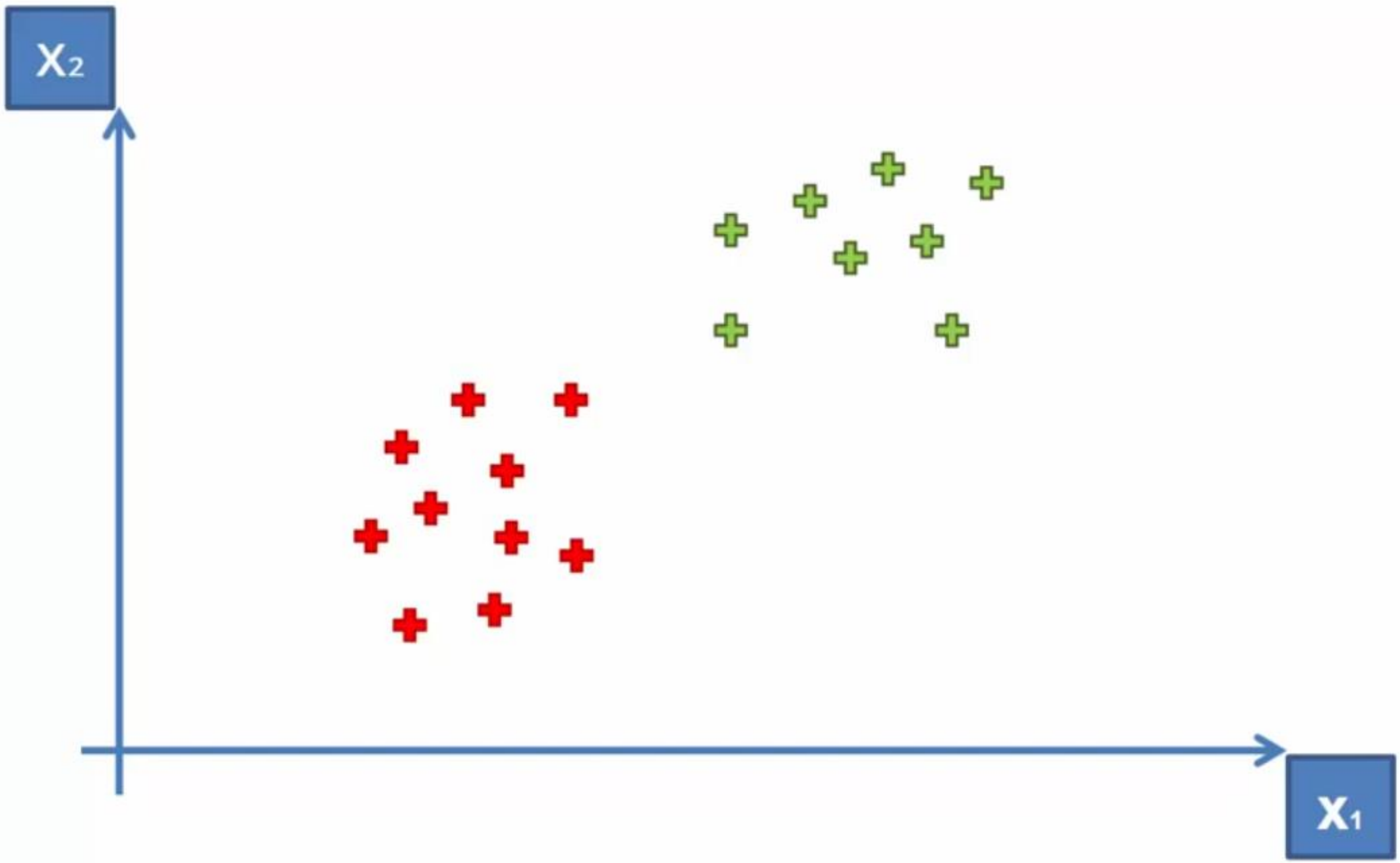
Classification

Let's refresh some points:
The goal of the classification is to find the decision boundary that best separates the points



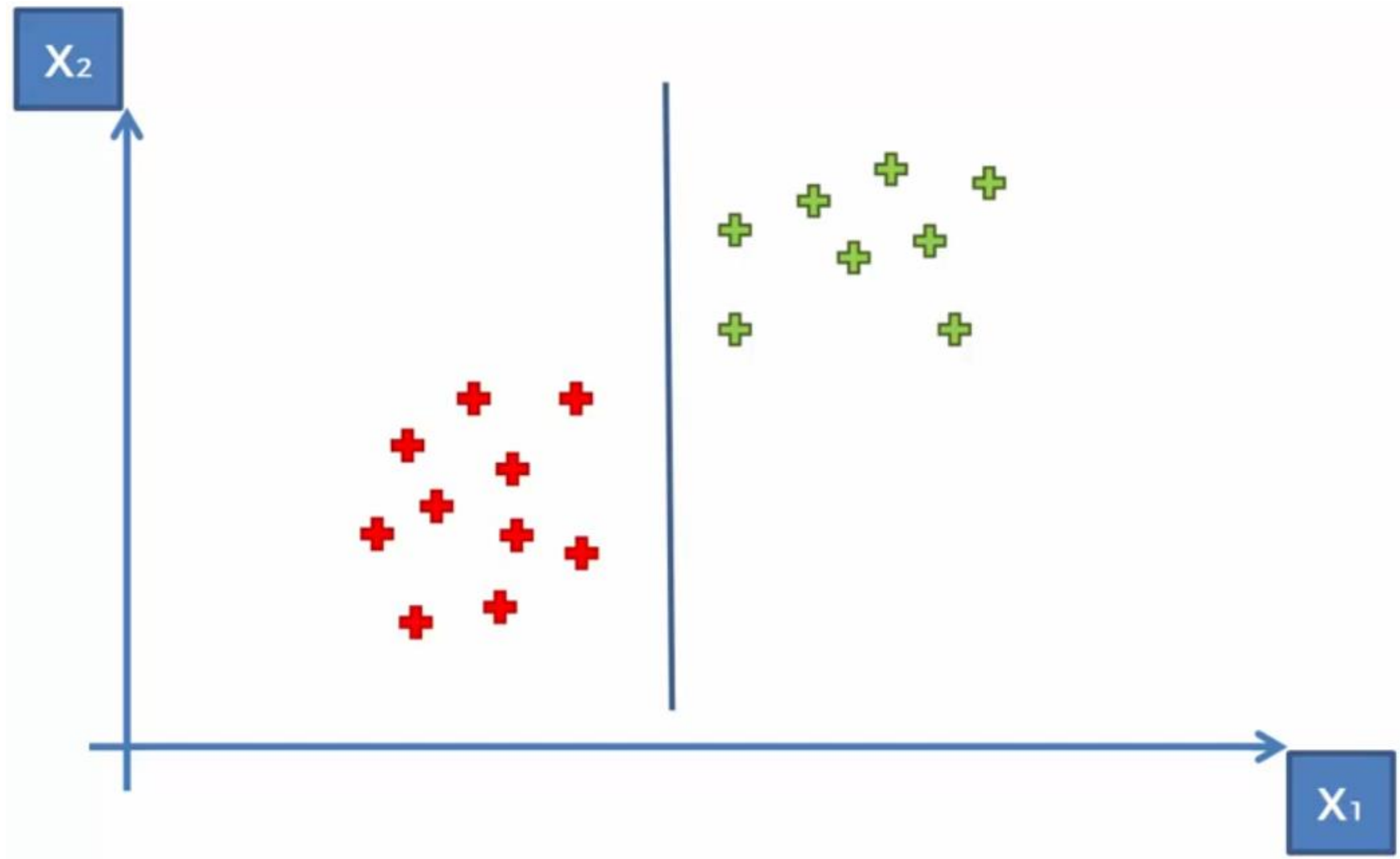
Classification

HOW?



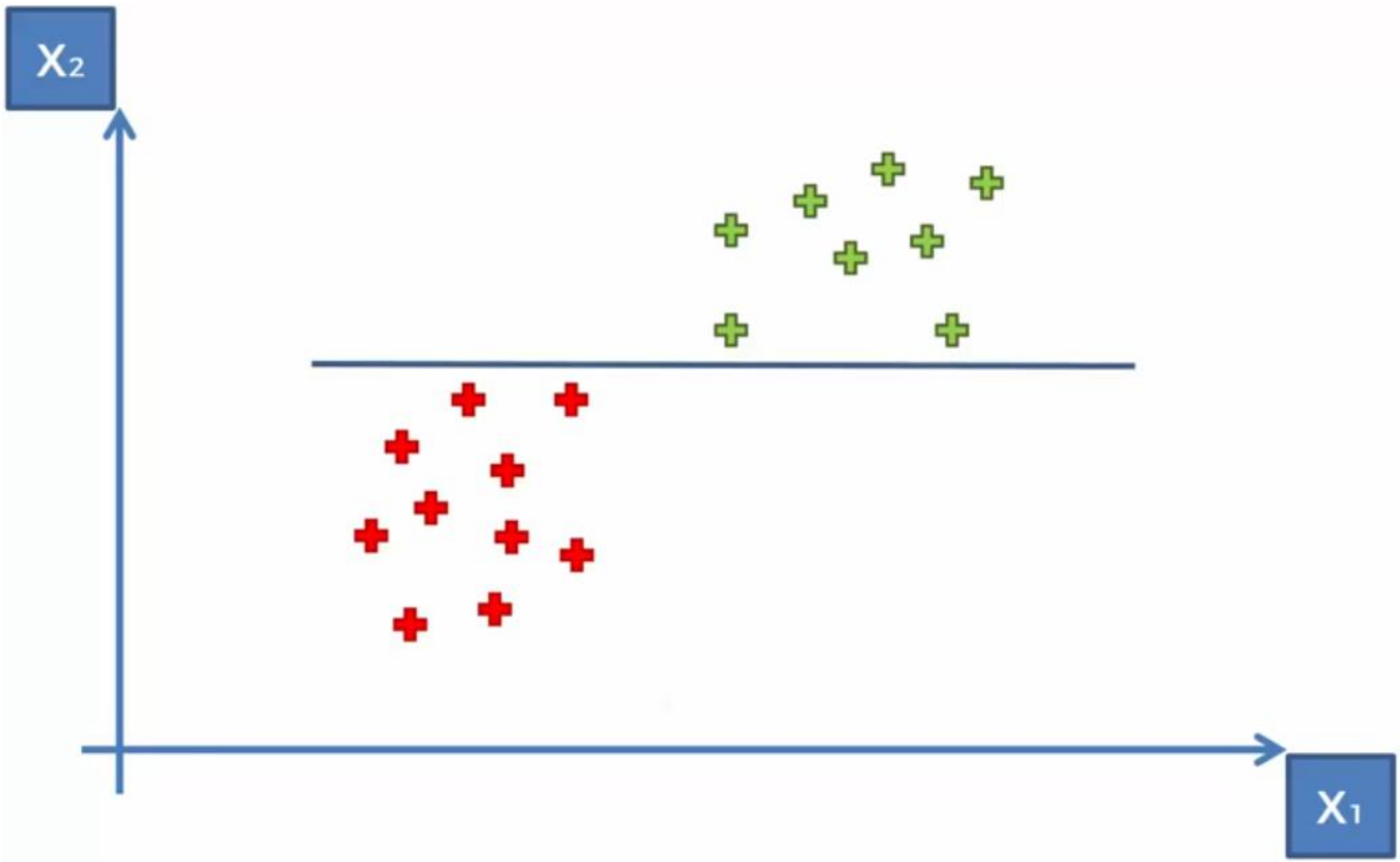
Classification

Here's the one way…



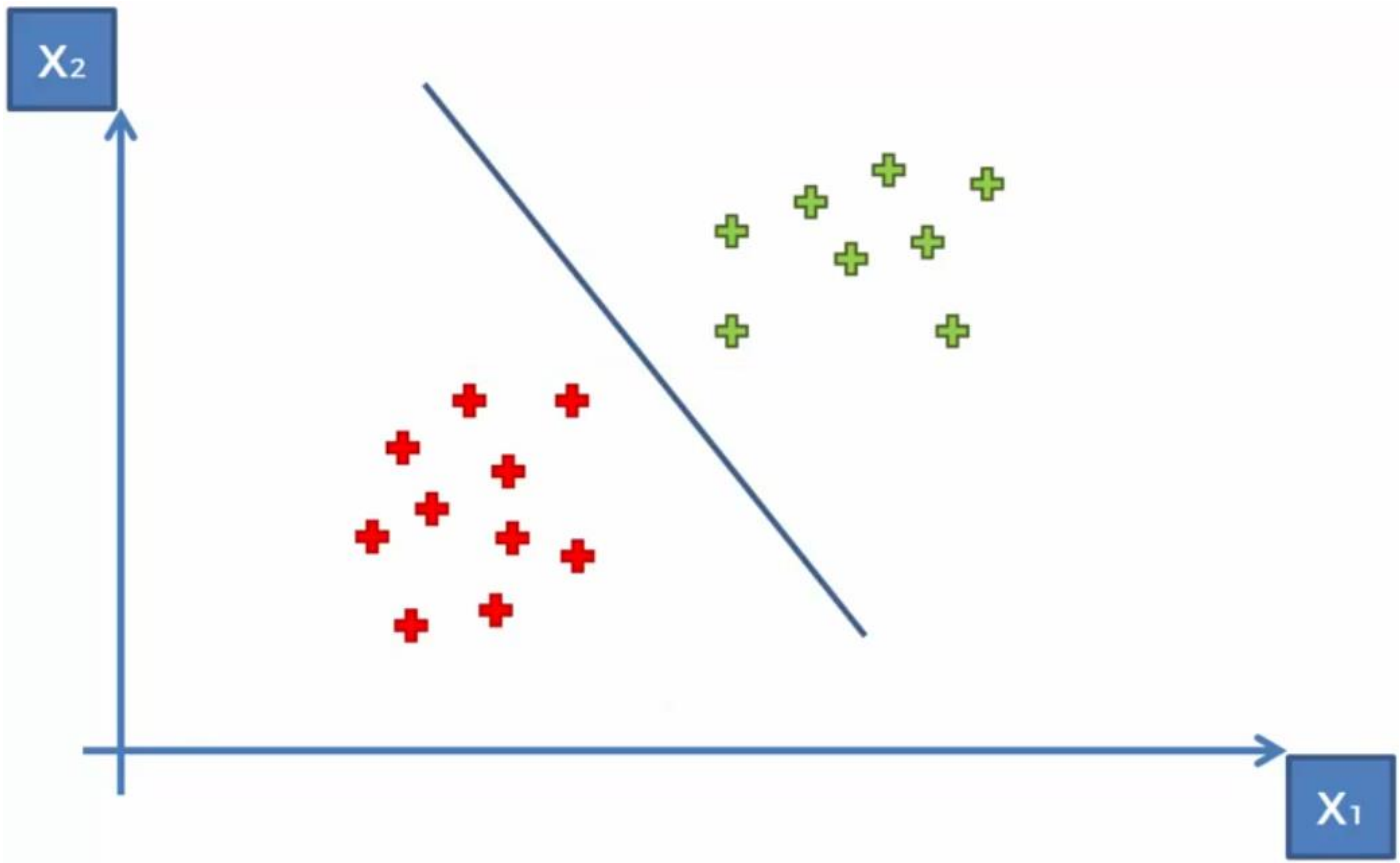
Classification

Here's another way…



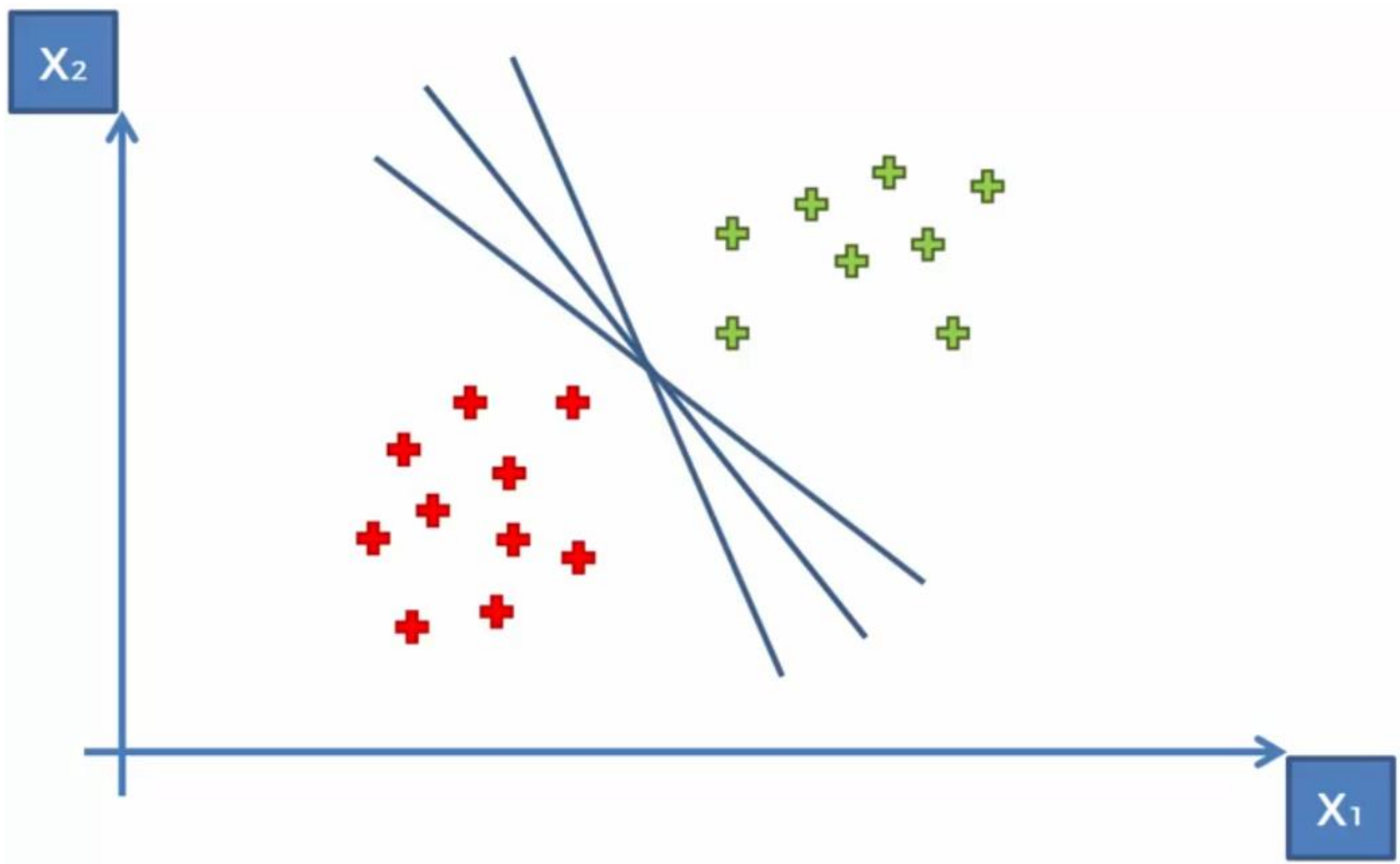
Classification

Keep going …

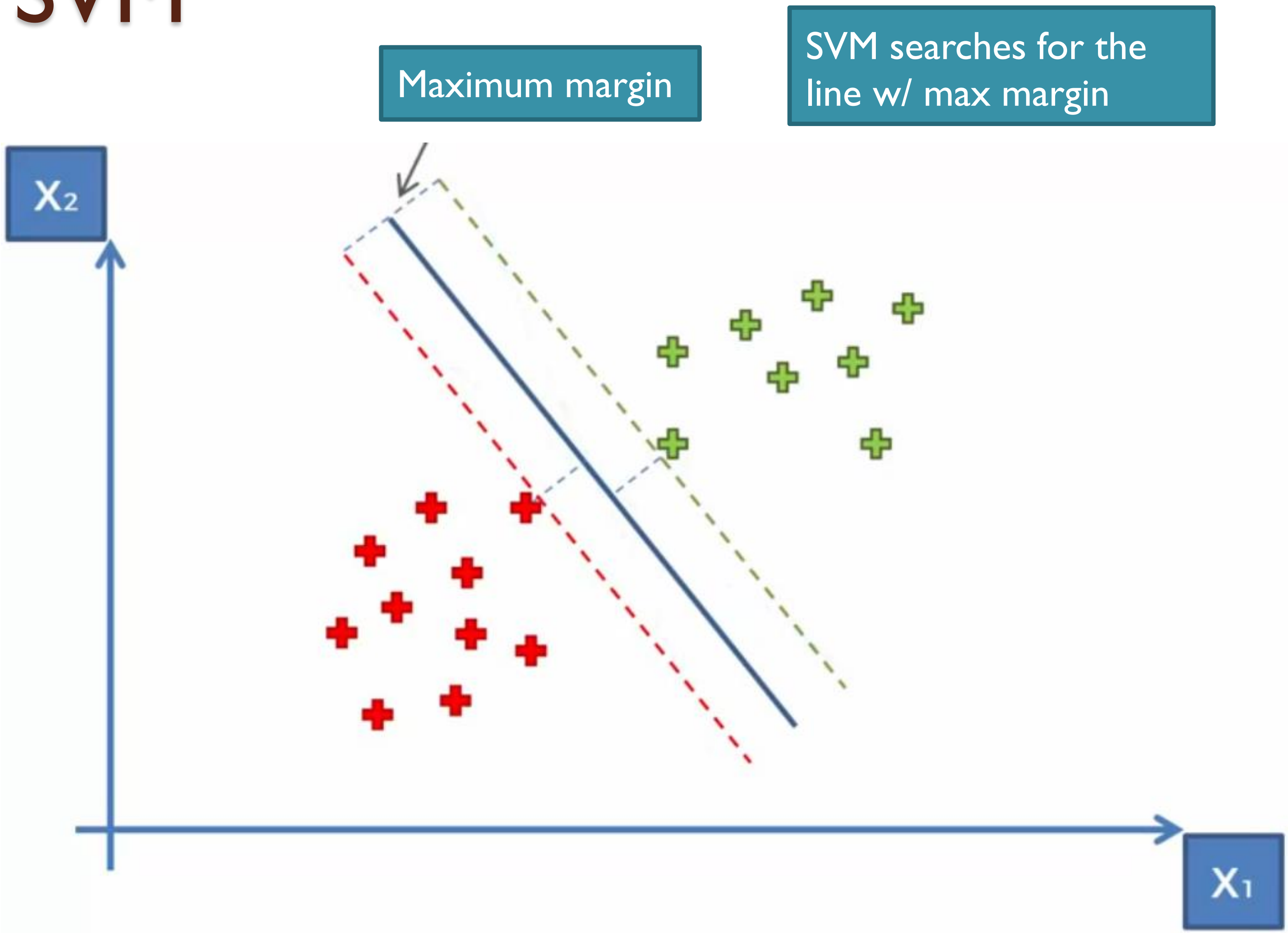


Classification

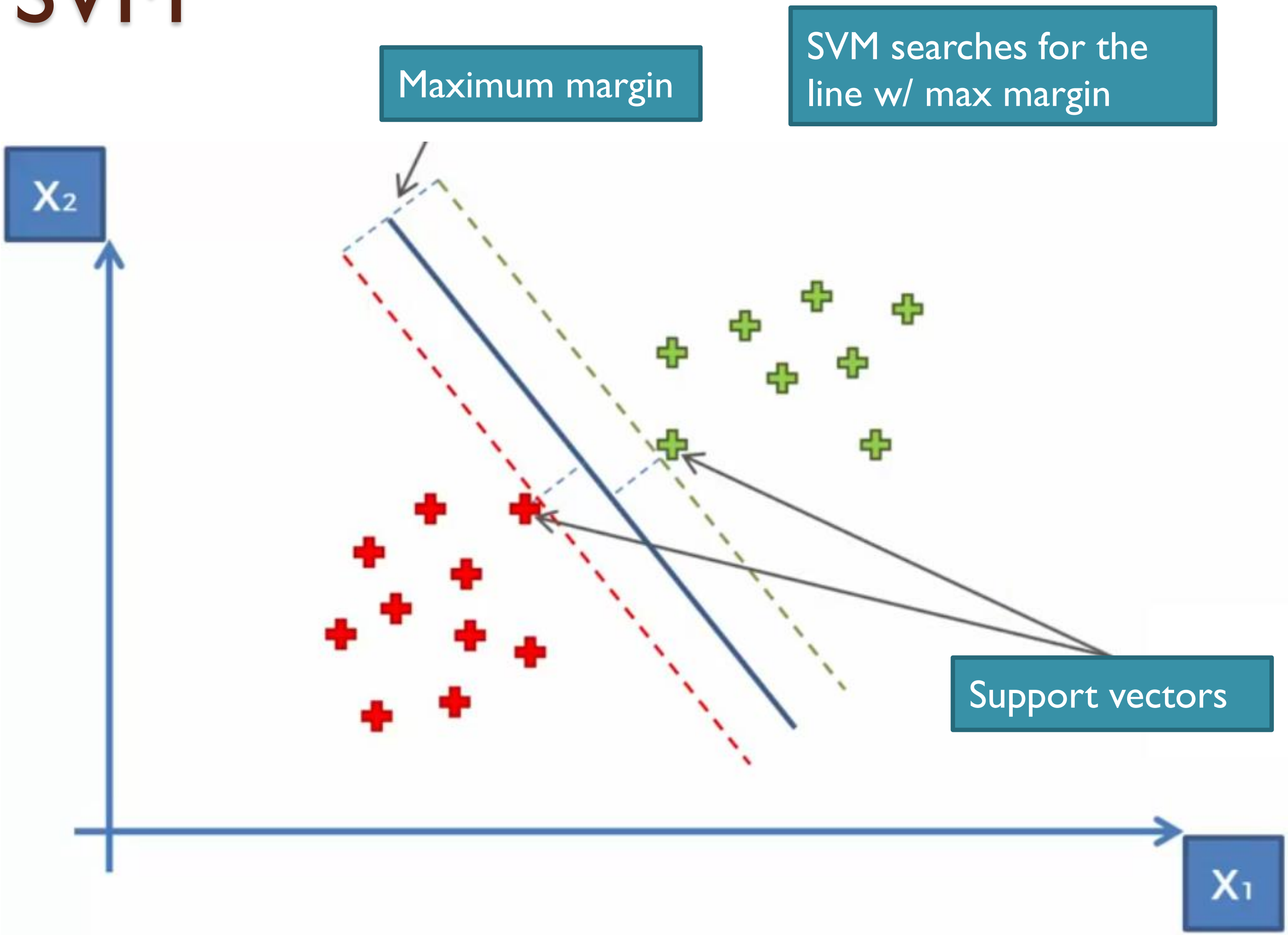
And more…We can actually do it infinitely



SVM

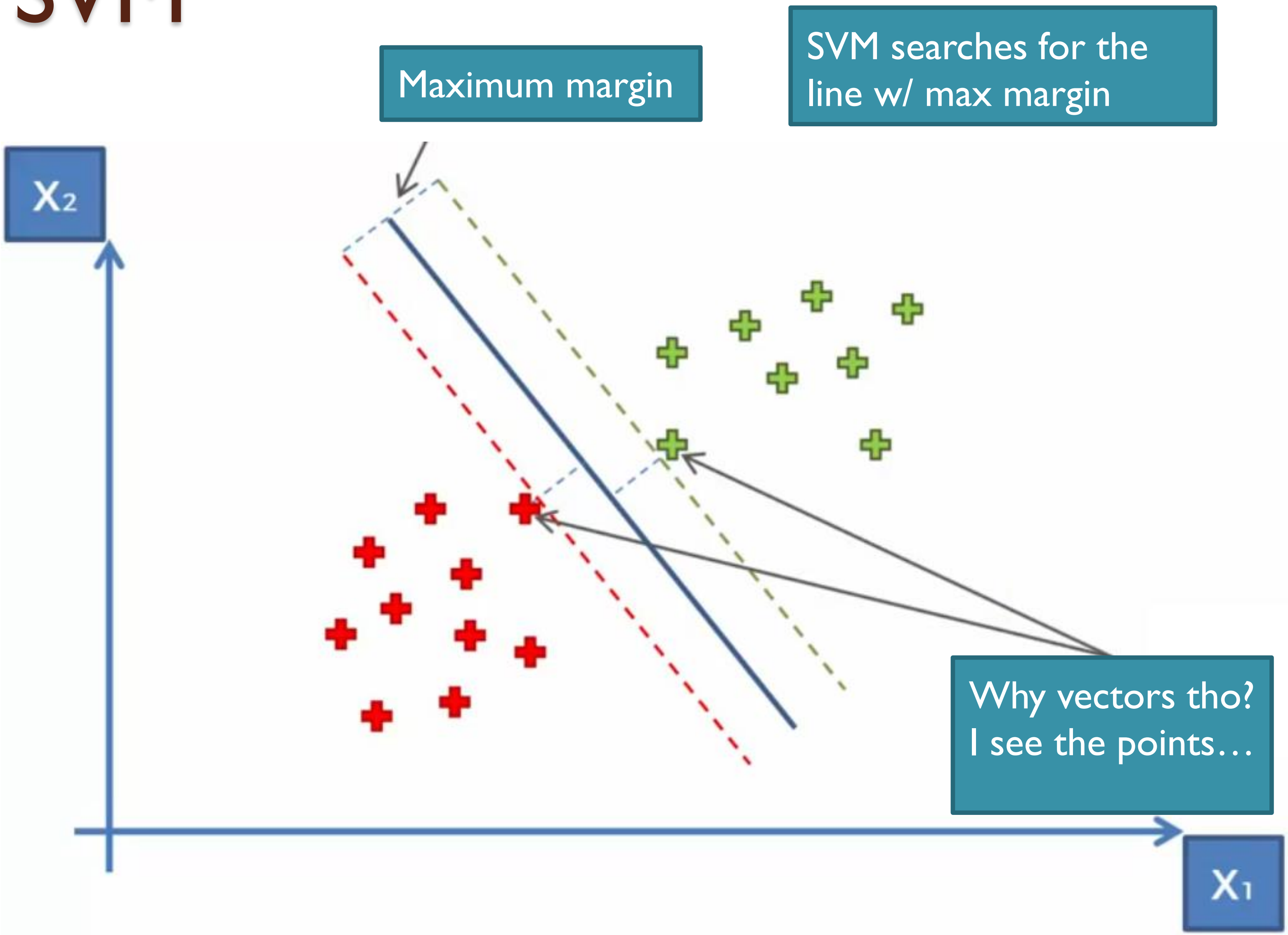


SVM

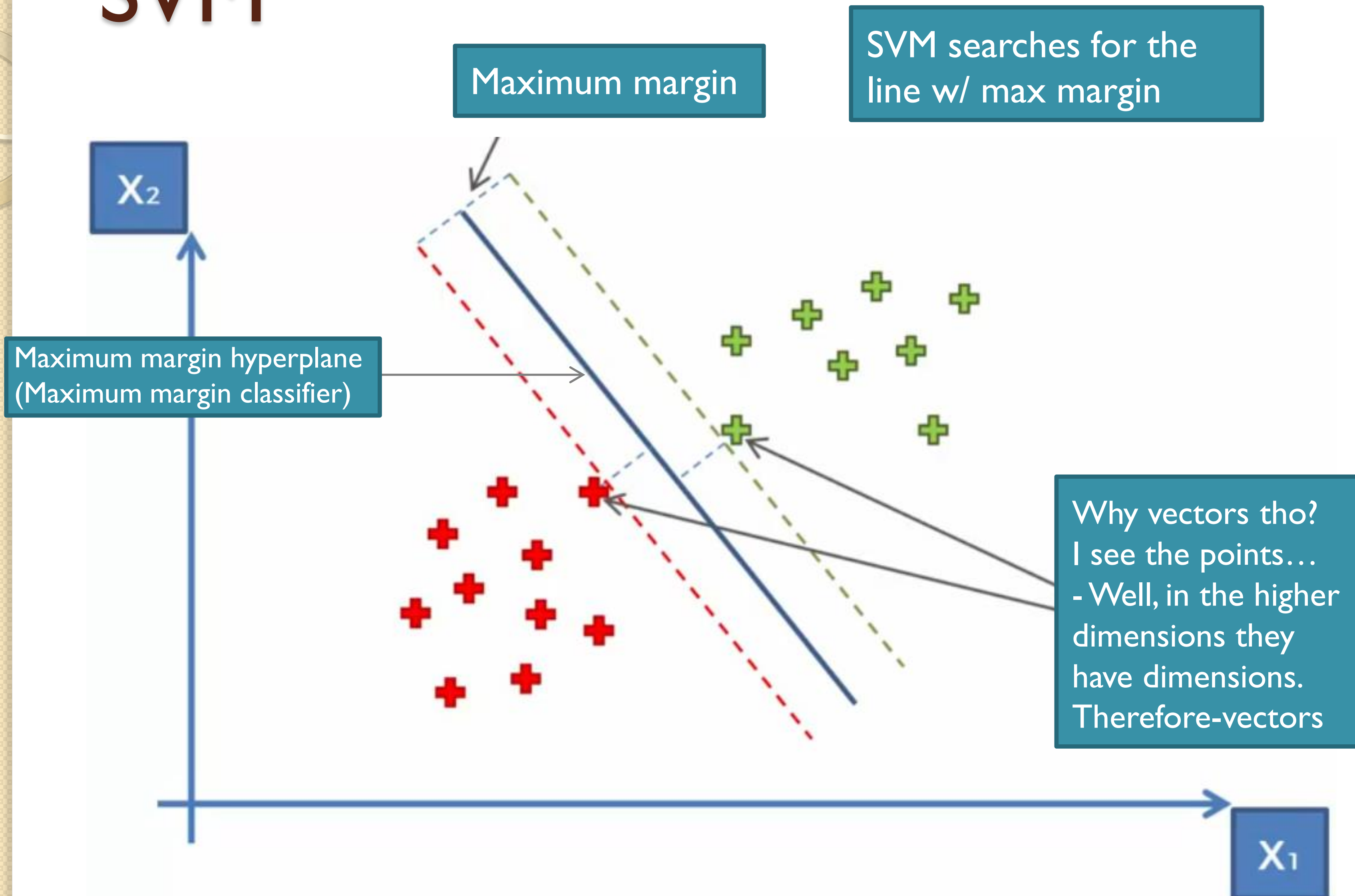


Maximum margin
SVM searches for the line w/ max margin
X2
X1
Support vectors

SVM


Maximum margin
SVM searches for the line w/ max margin
X2
Why vectors tho? I see the points…
X1

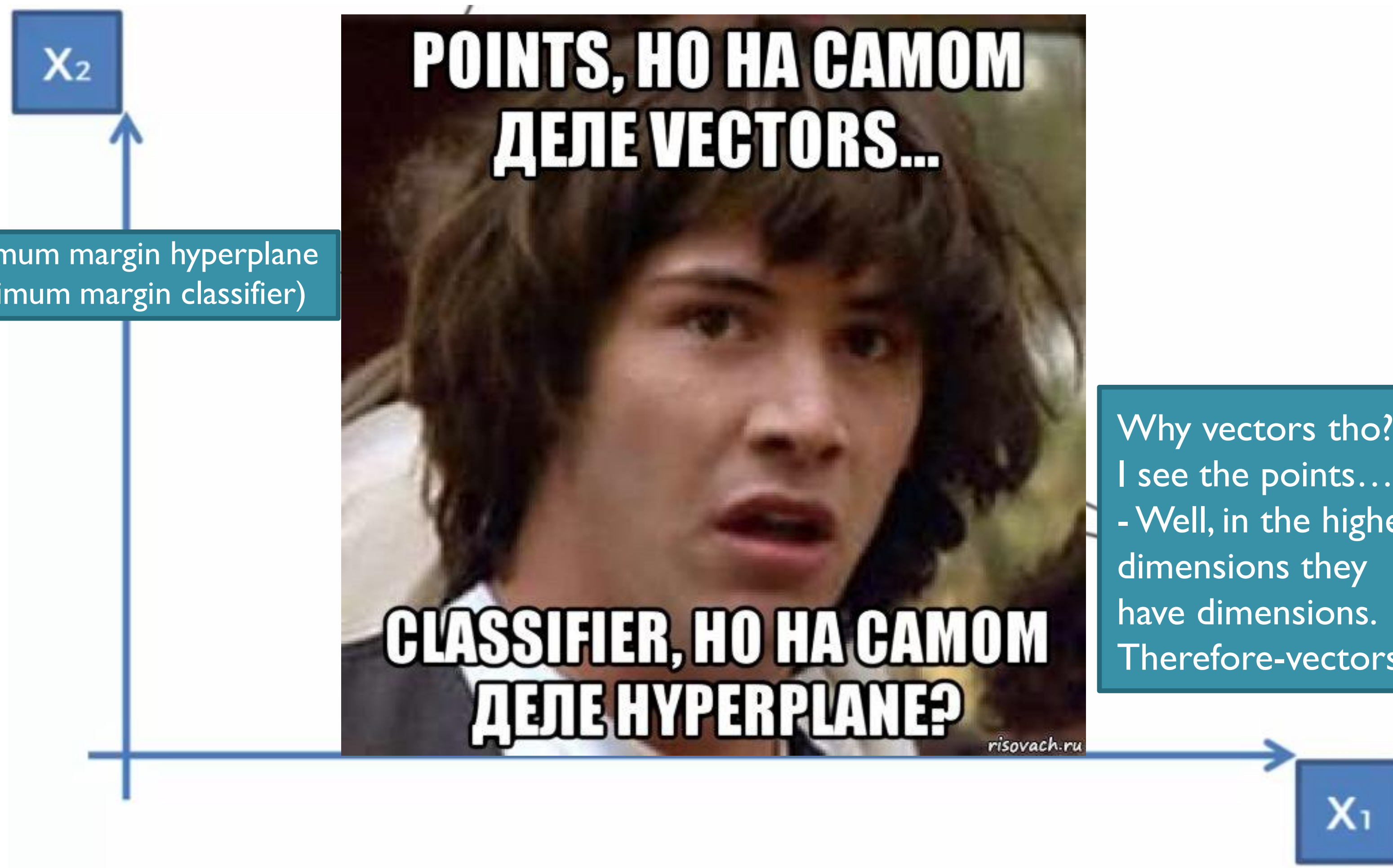
SVM


Maximum margin
SVM searches for the line w/ max margin
X_2
Maximum margin hyperplane (Maximum margin classifier)
Why vectors tho?
I see the points…
- Well, in the higher dimensions they have dimensions.
Therefore-vectors
X_1

SVM

Maximum margin

SVM searches for the line w/ max margin



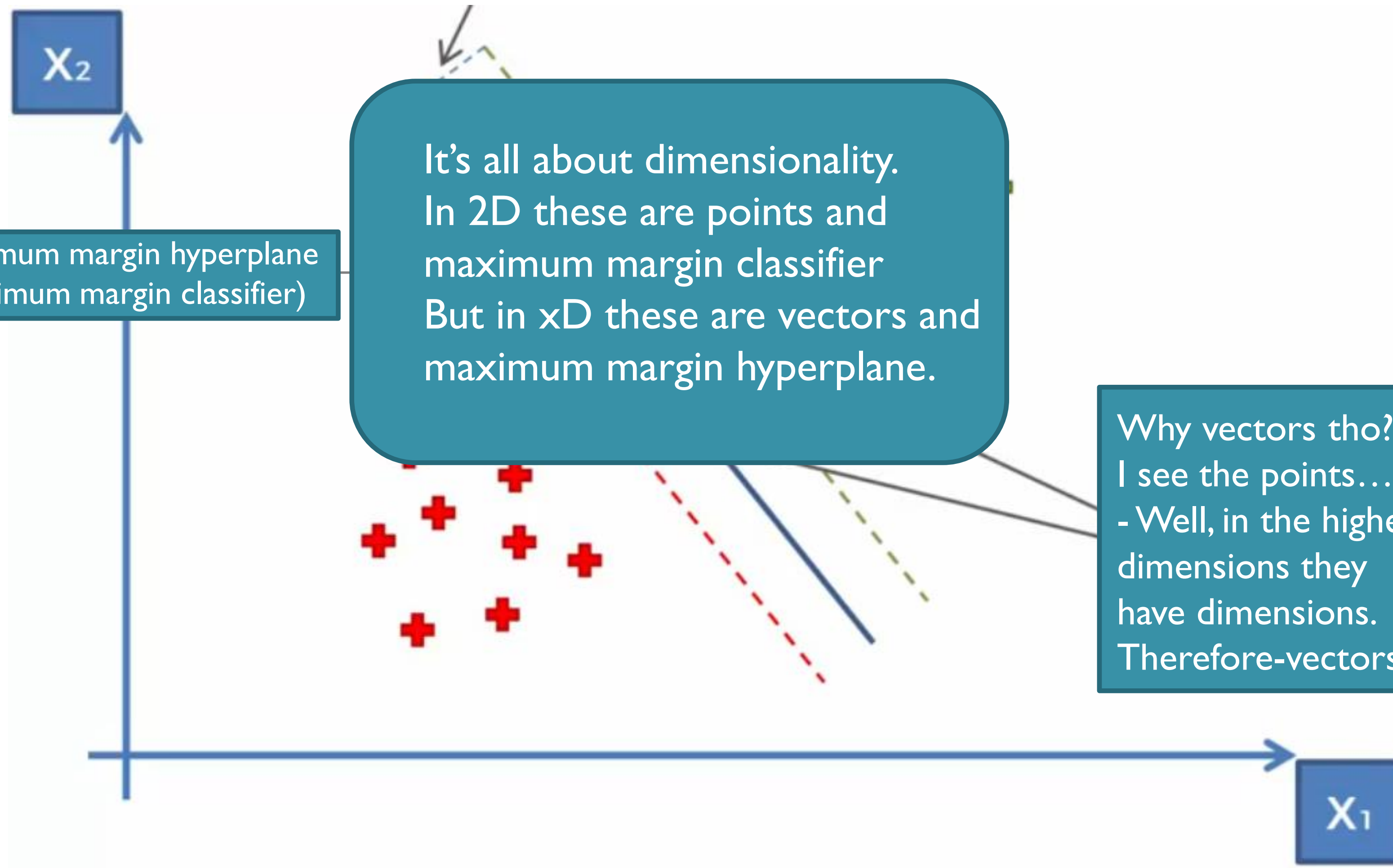
Maximum margin hyperplane
(Maximum margin classifier)

Why vectors tho?
I see the points…
- Well, in the higher dimensions they have dimensions. Therefore-vectors

SVM

Maximum margin

SVM searches for the line w/ max margin

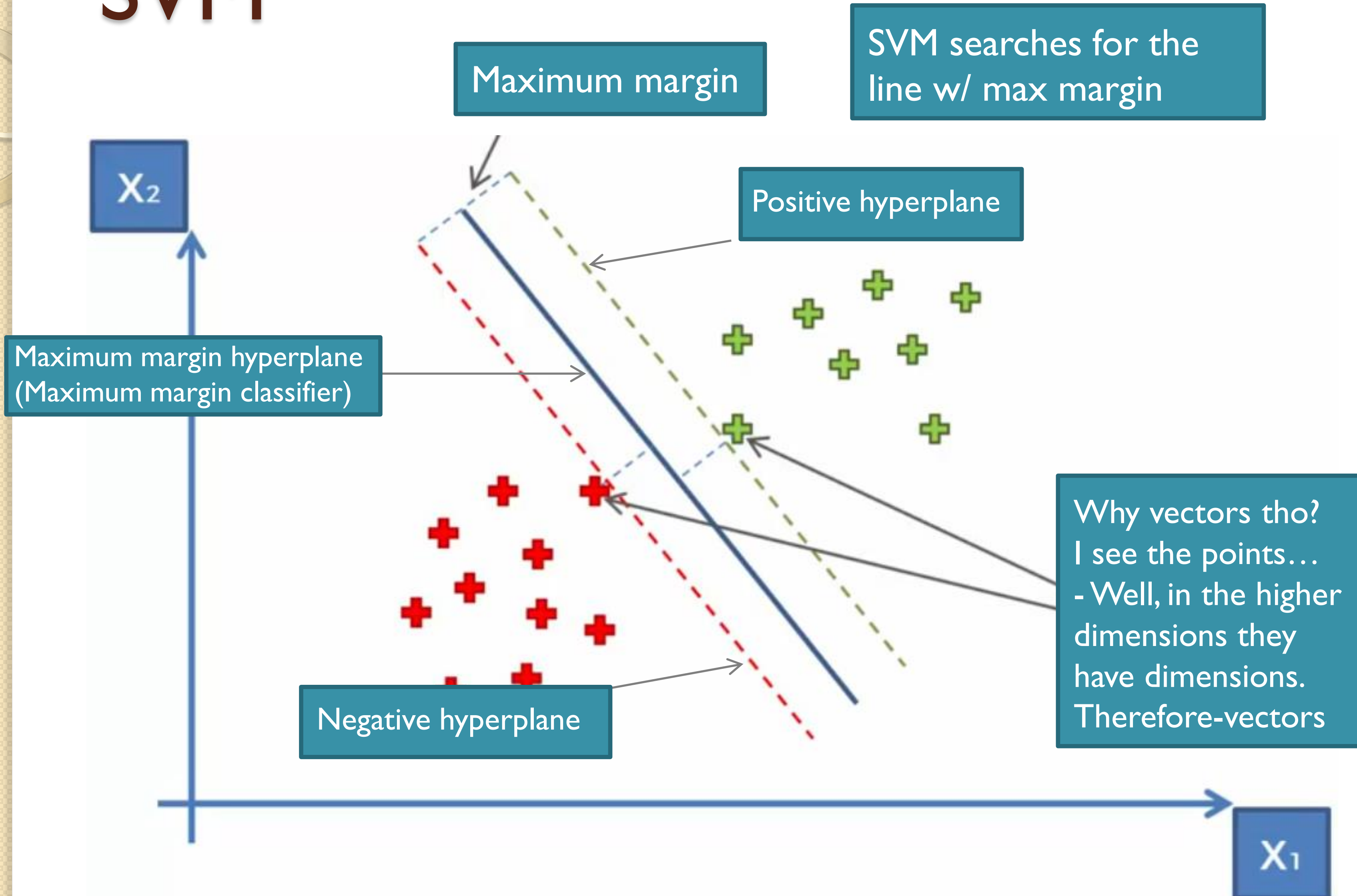


Maximum margin hyperplane
(Maximum margin classifier)

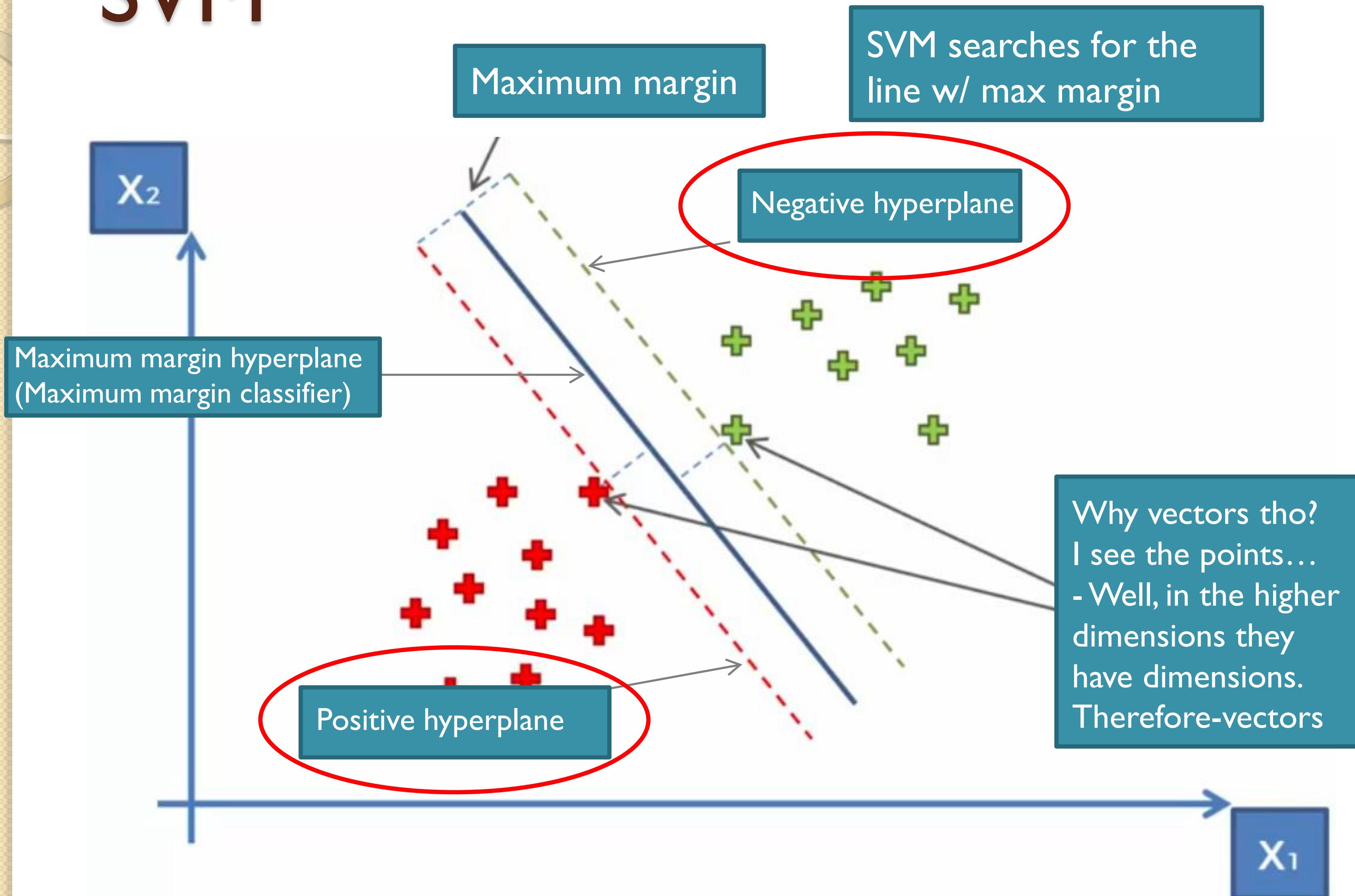
It's all about dimensionality.
In 2D these are points and maximum margin classifier
But in xD these are vectors and maximum margin hyperplane.

Why vectors tho?
I see the points…
- Well, in the higher dimensions they have dimensions.
Therefore-vectors

SVM

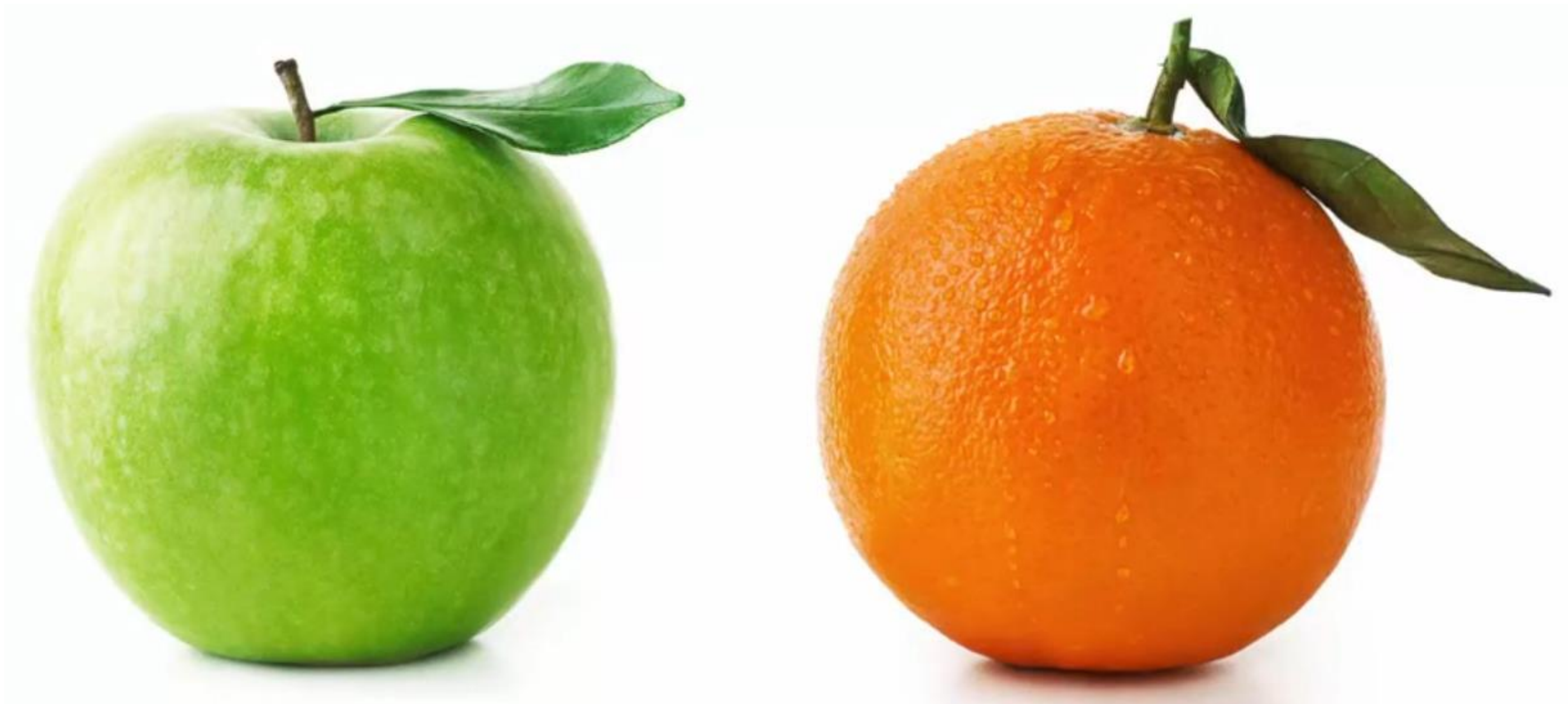

Maximum margin
SVM searches for the line w/ max margin
X2
Positive hyperplane
Maximum margin hyperplane (Maximum margin classifier)
Why vectors tho? I see the points… - Well, in the higher dimensions they have dimensions. Therefore-vectors
Negative hyperplane
X1

SVM


Maximum margin
SVM searches for the line w/ max margin
X_2
Negative hyperplane
Maximum margin hyperplane (Maximum margin classifier)
Why vectors tho?
I see the points…
- Well, in the higher dimensions they have dimensions. Therefore-vectors
Positive hyperplane
X_1

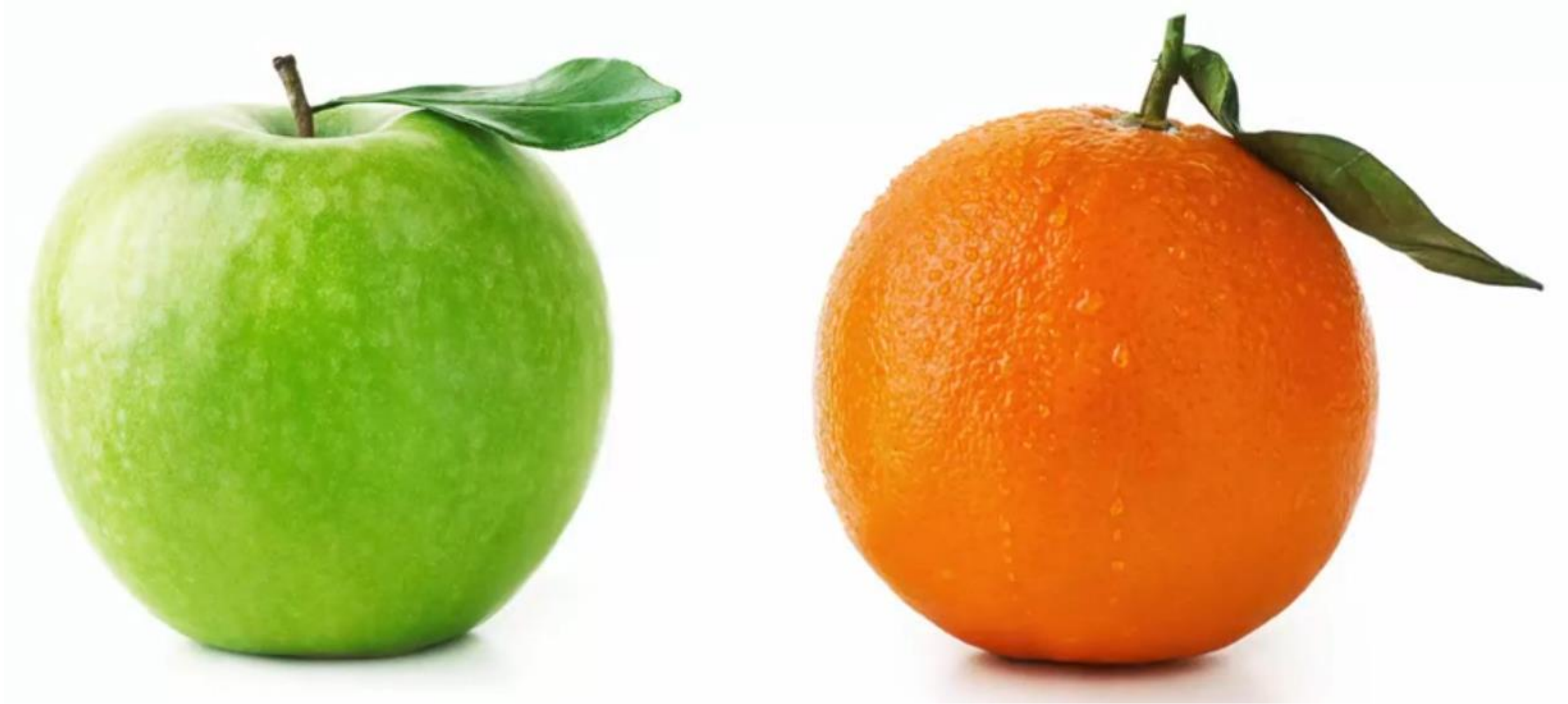
So what's that special about SVM?

SVM



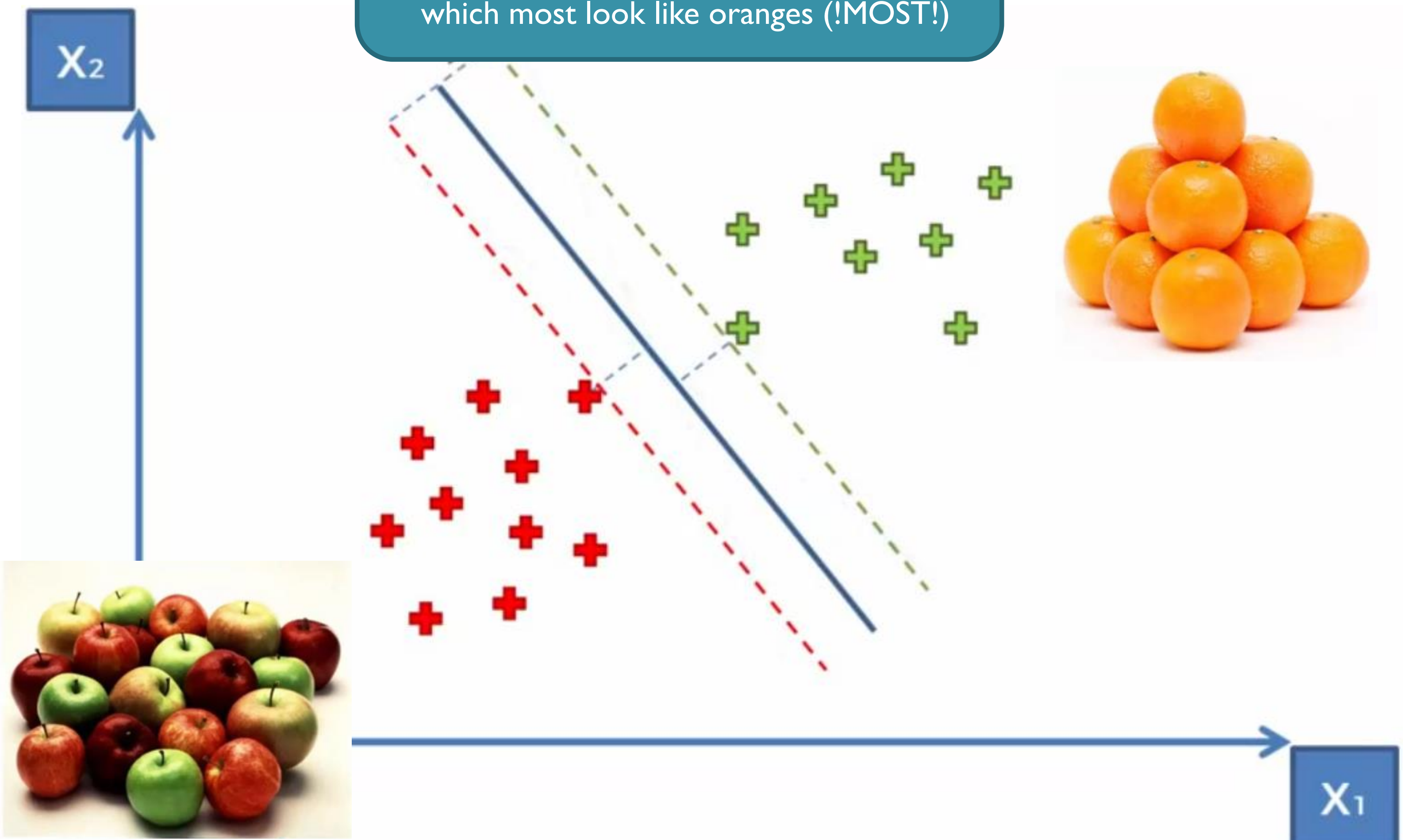
SVM

Let’s say you wan to teach the model to classify oranges and apples



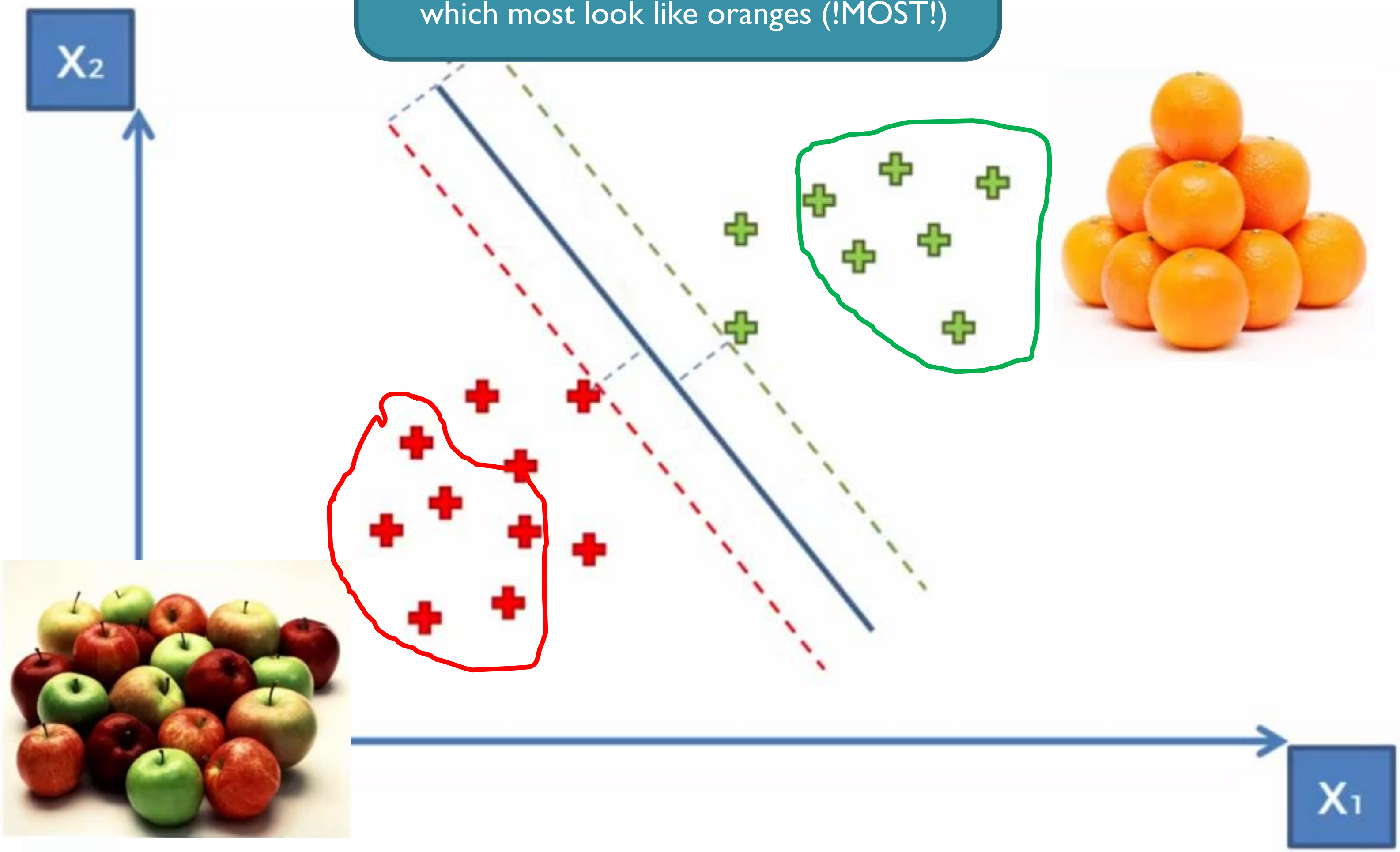
SVM

What the most algorithm will do?
They will search for the points which most look like apples and the points which most look like oranges (!MOST!)



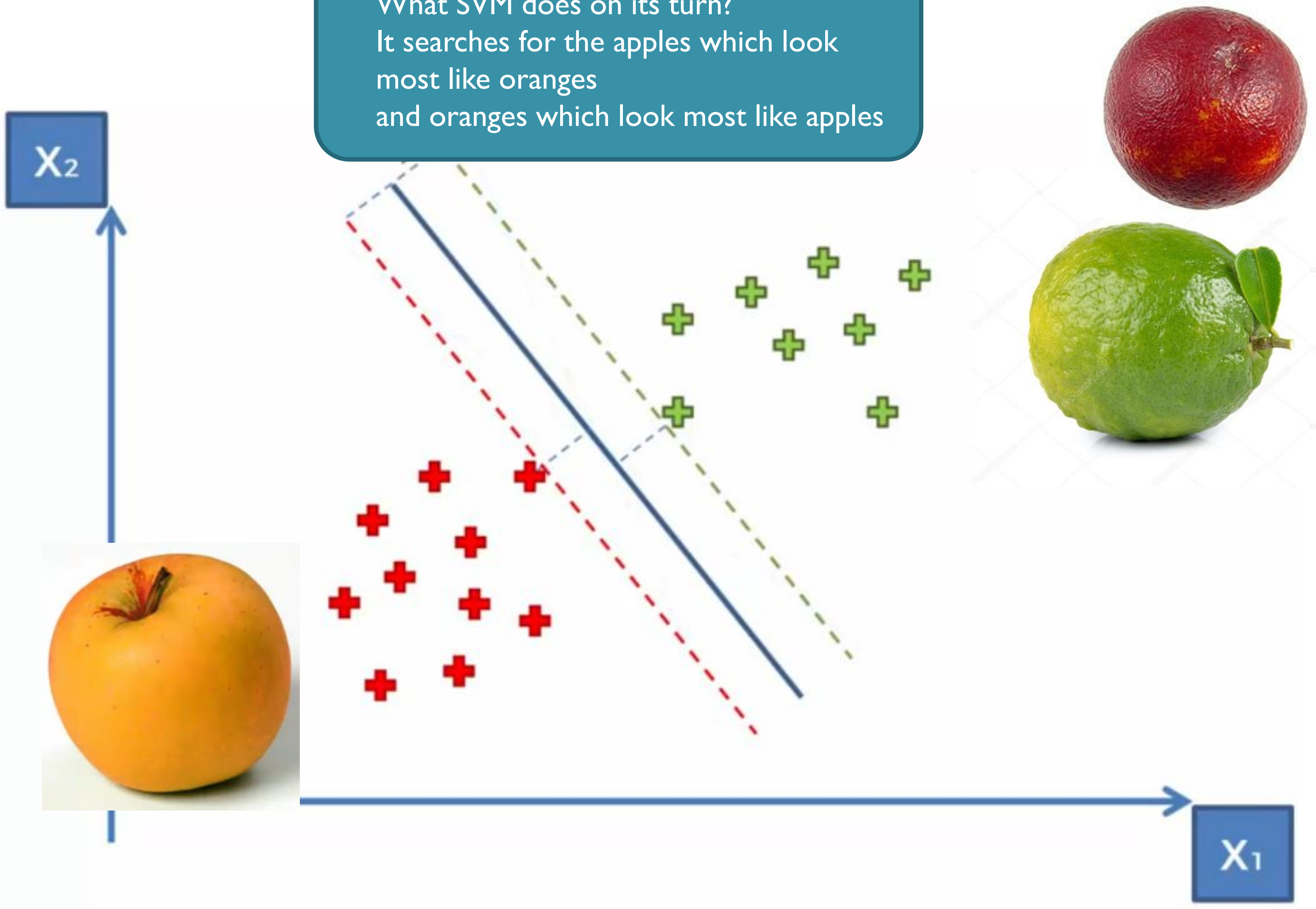
SVM

What the most algorithm will do?
They will search for the points which most look like apples and the points which most look like oranges (!MOST!)



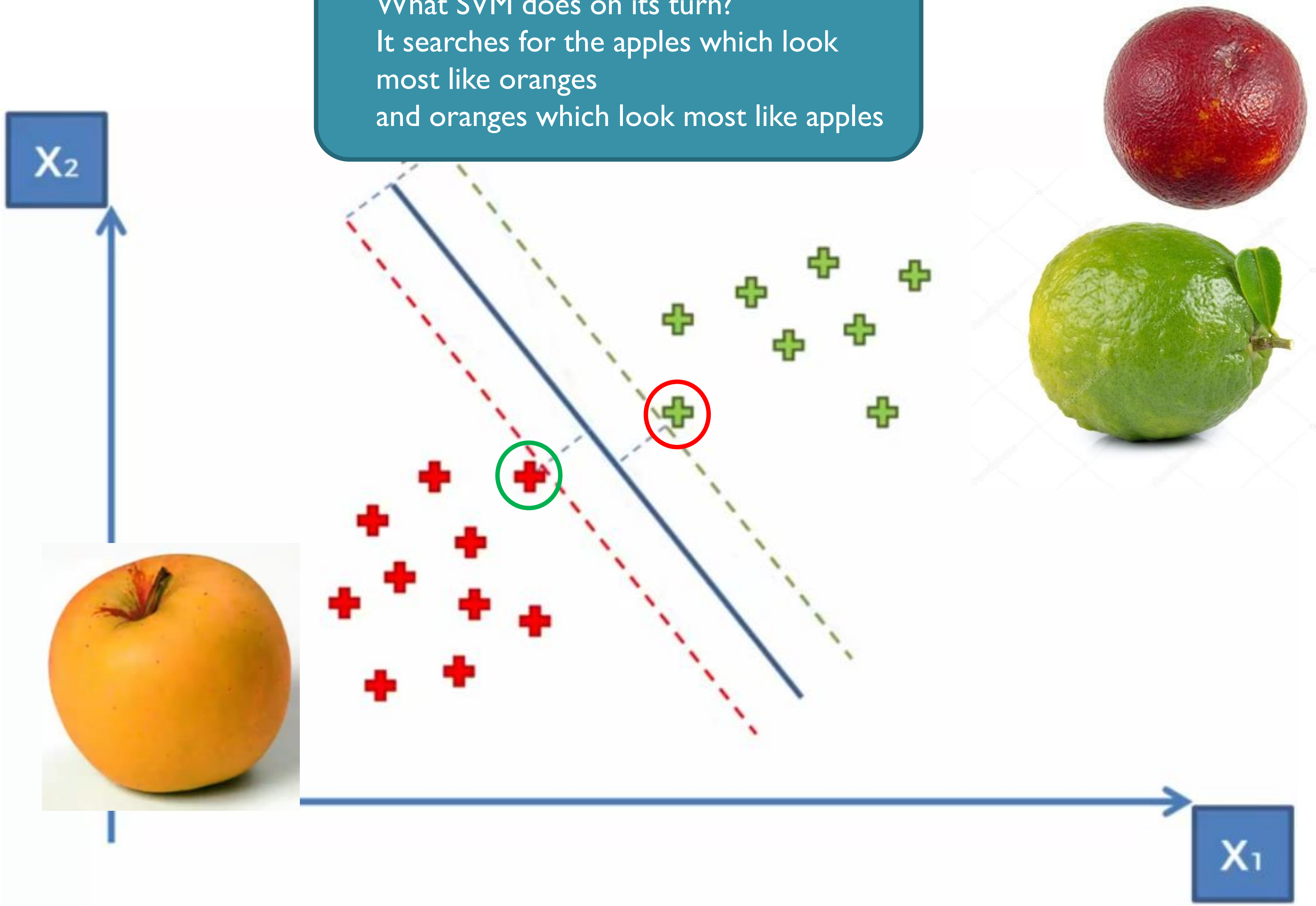
SVM

What SVM does on its turn?
It searches for the apples which look most like oranges
and oranges which look most like apples



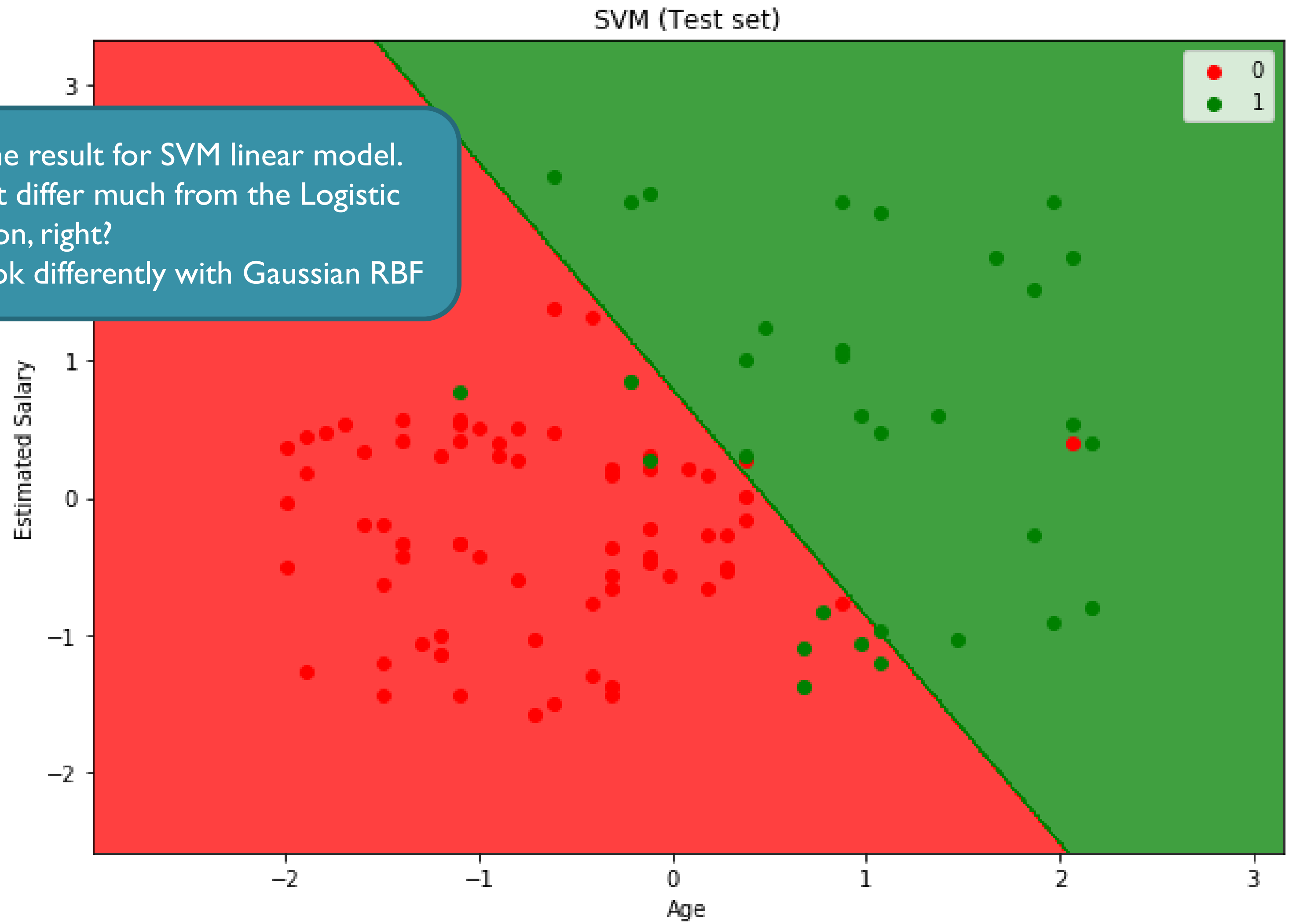
SVM

What SVM does on its turn?
It searches for the apples which look most like oranges
and oranges which look most like apples

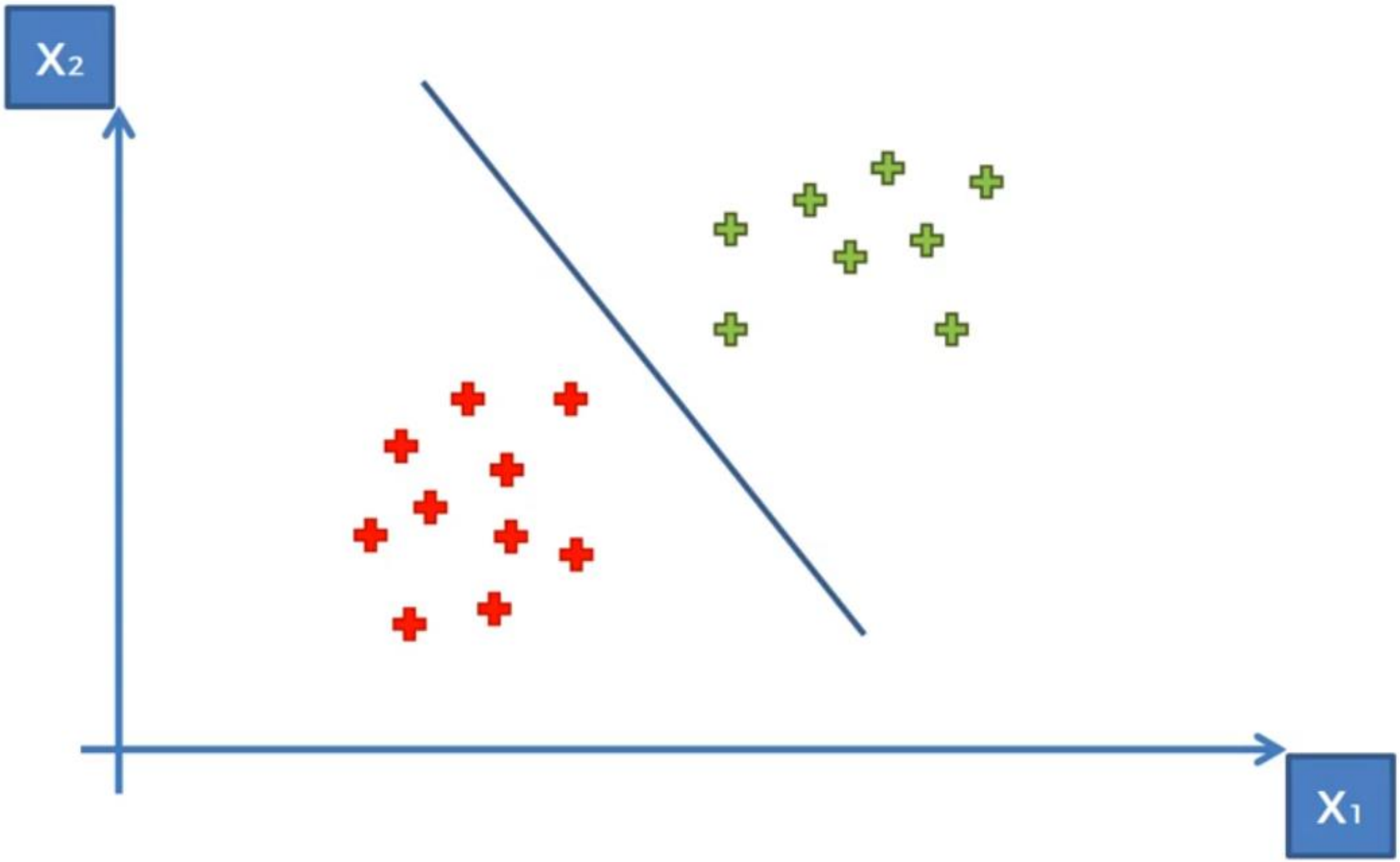


Lab's output

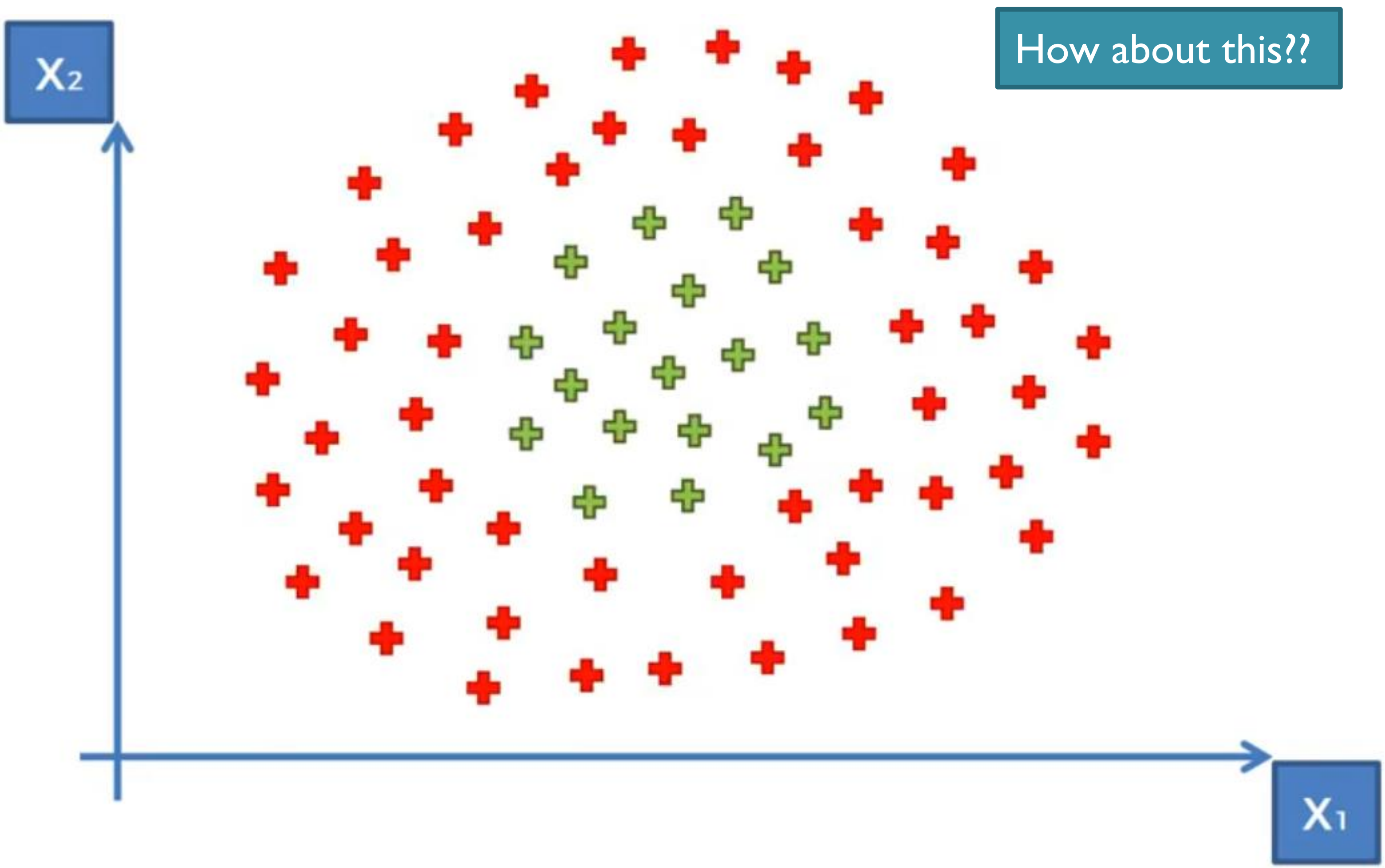
This is the result for SVM linear model.
It doesn't differ much from the Logistic Regression, right?
It will look differently with Gaussian RBF



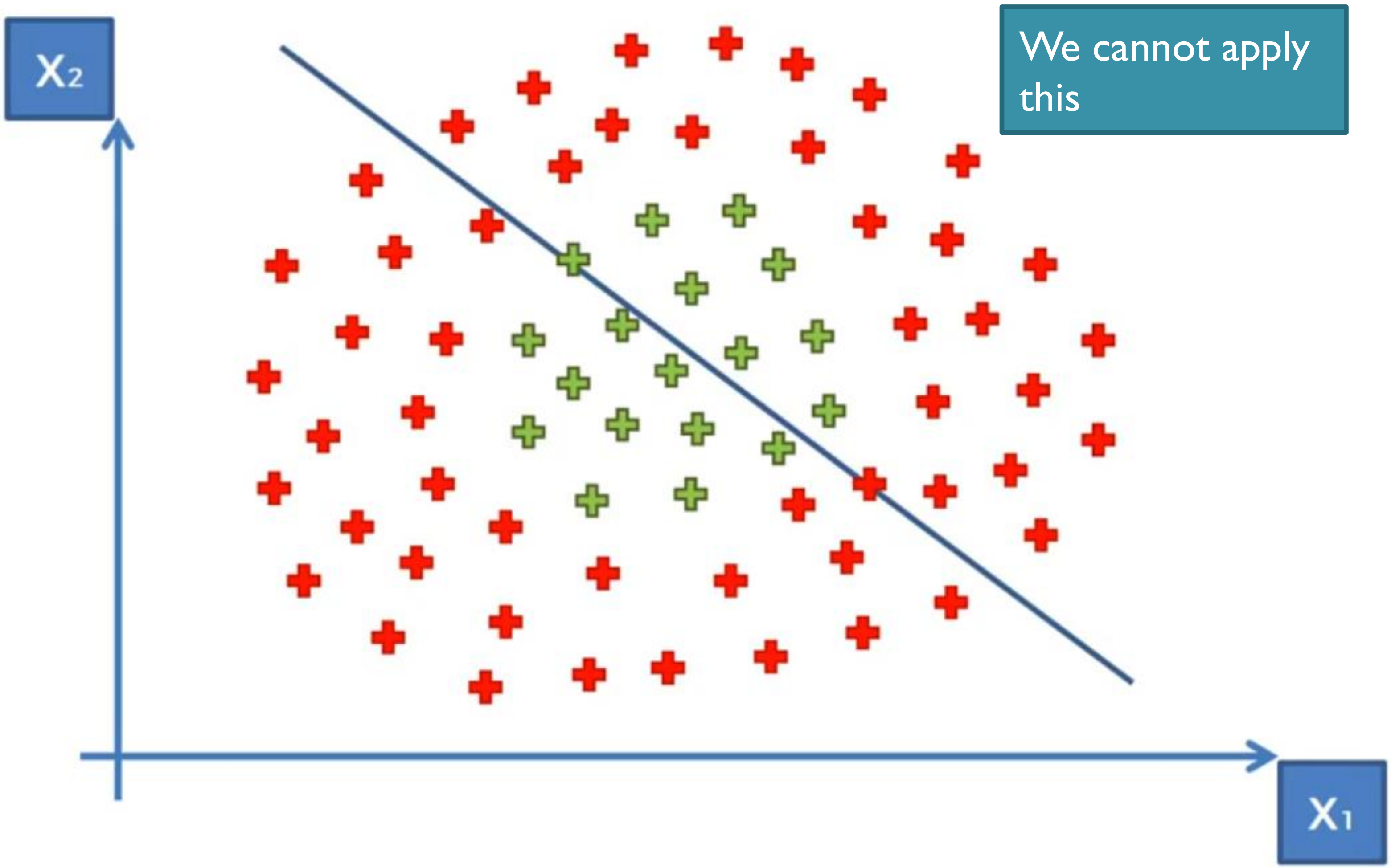
SVM. Part 2



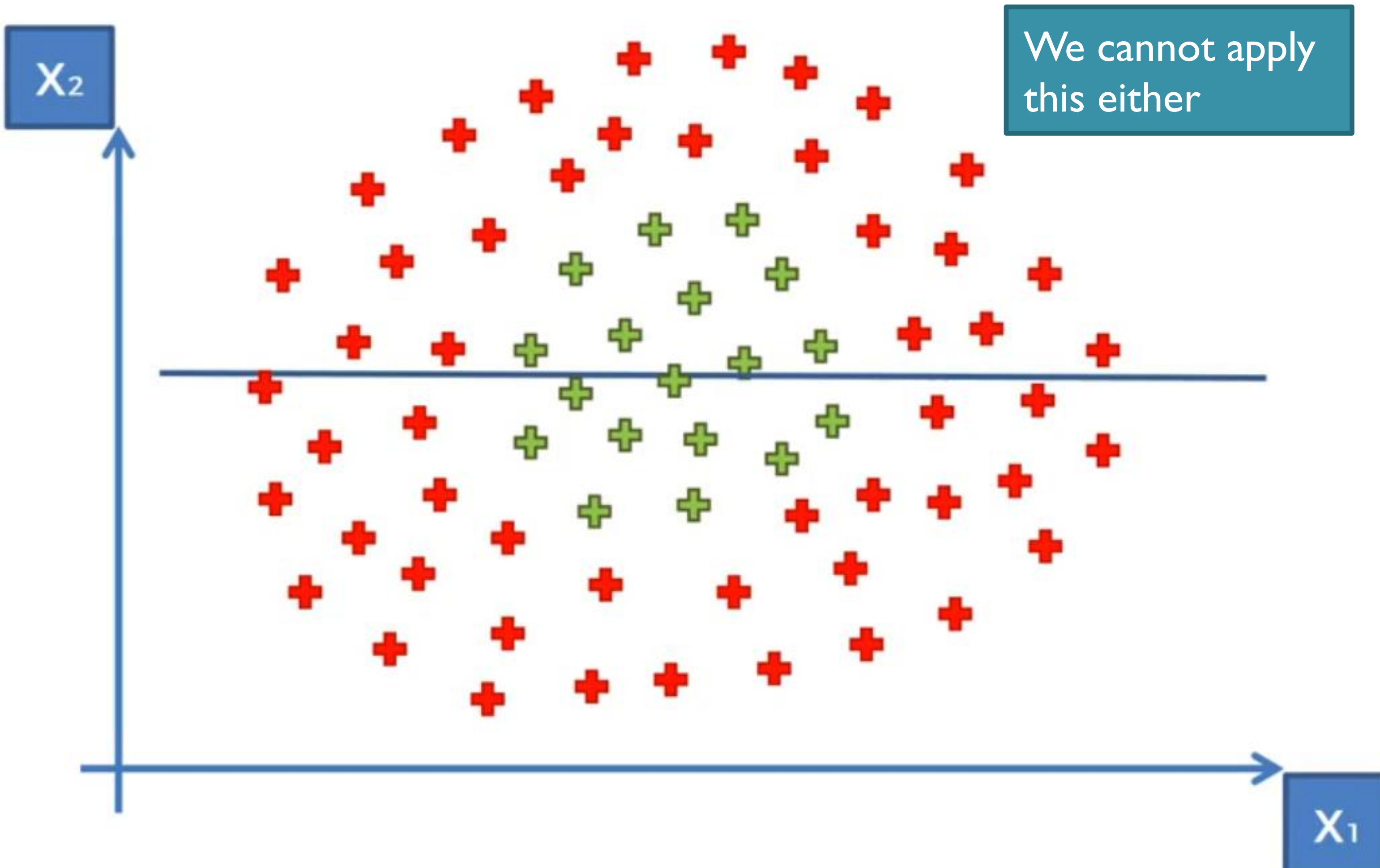
SVM. Part 2



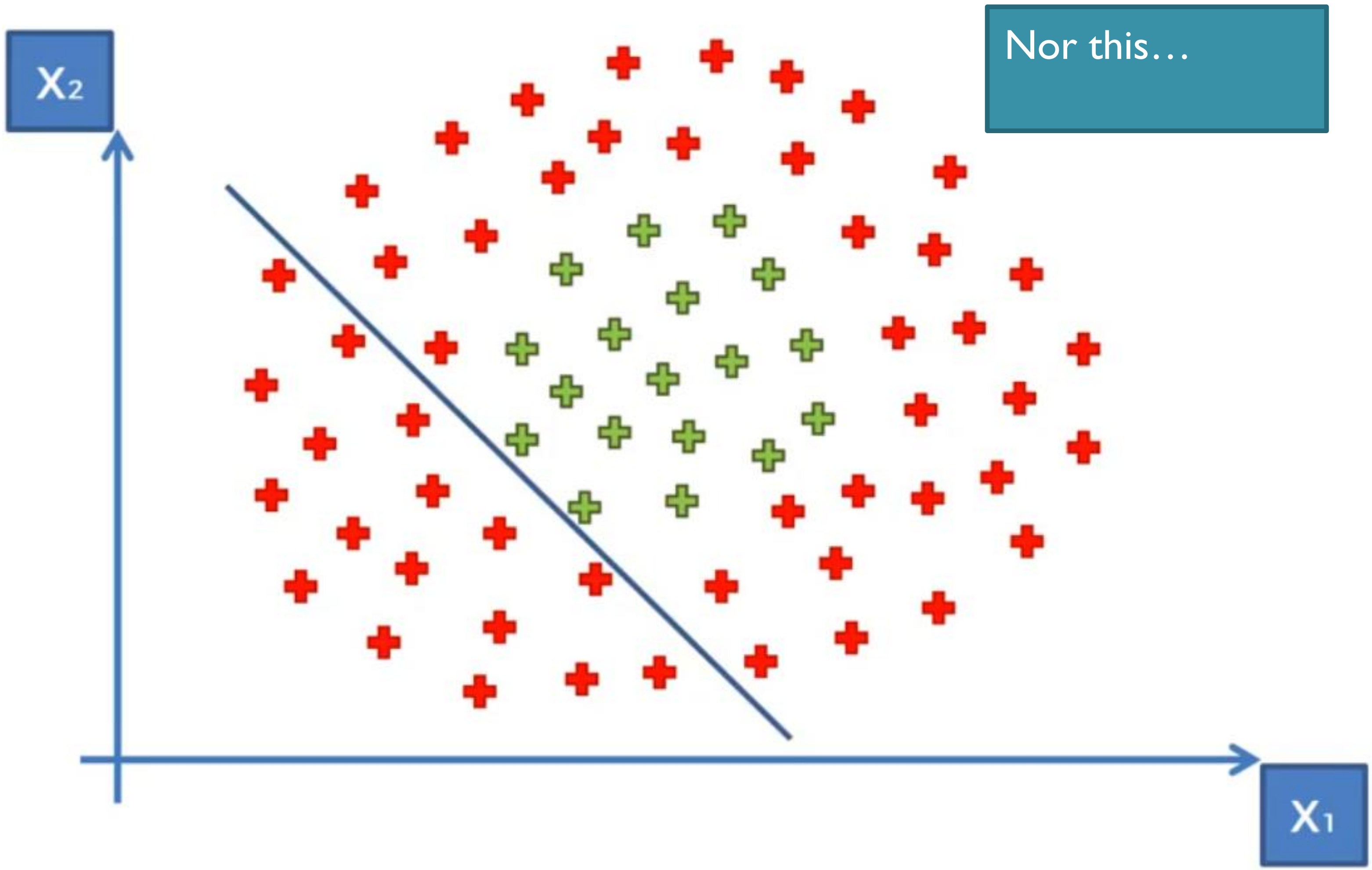
SVM. Part 2



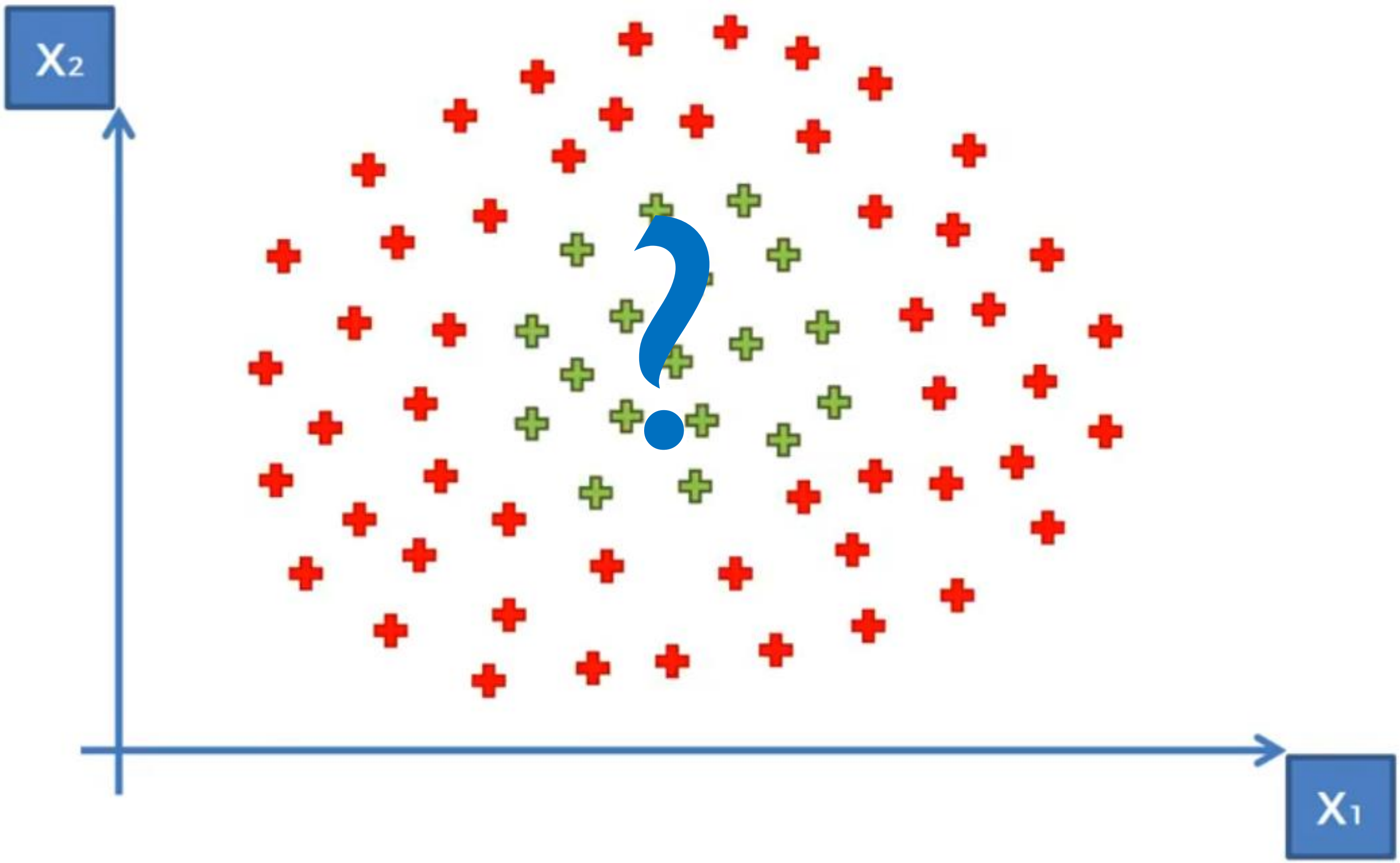
SVM. Part 2



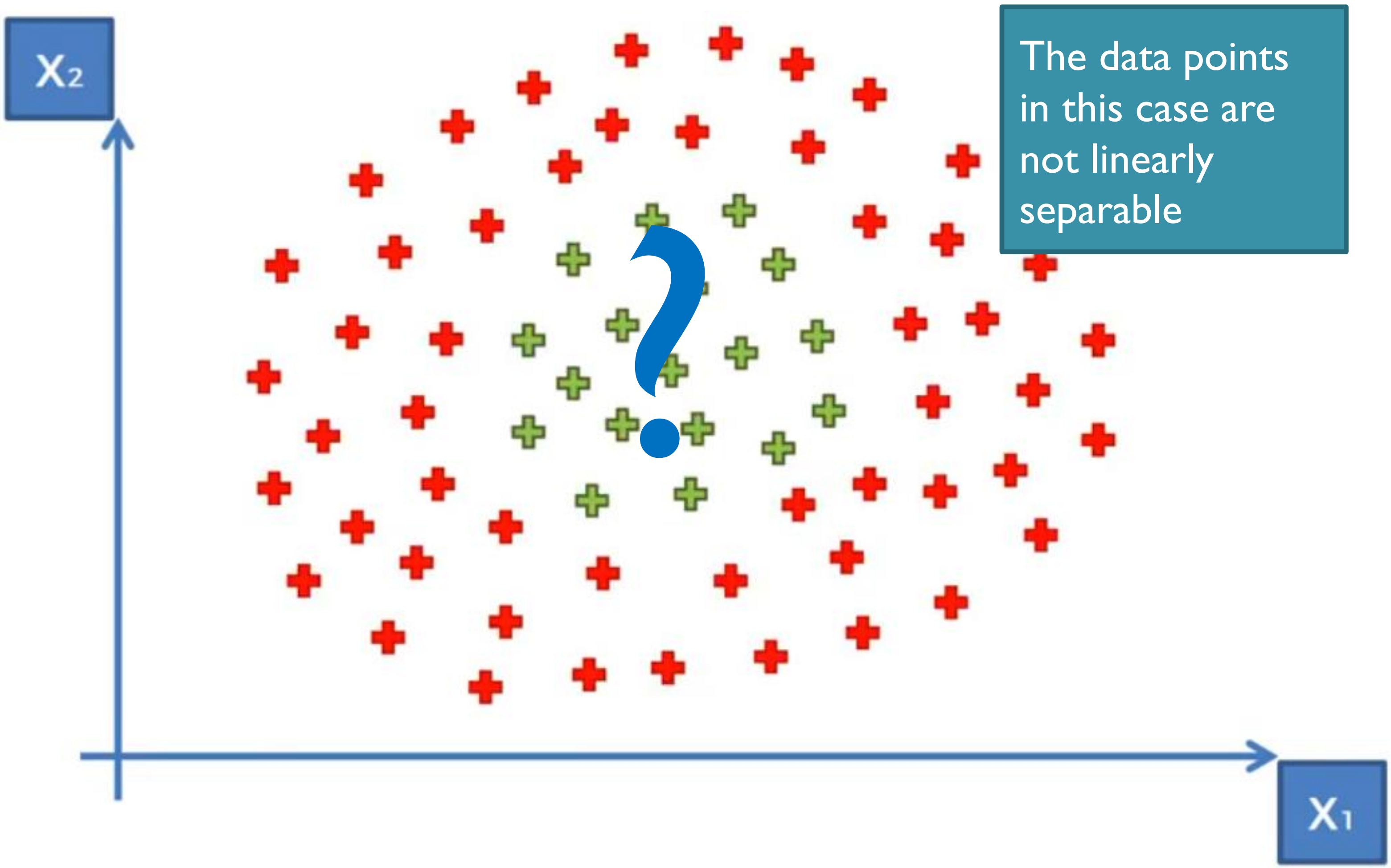
SVM. Part 2



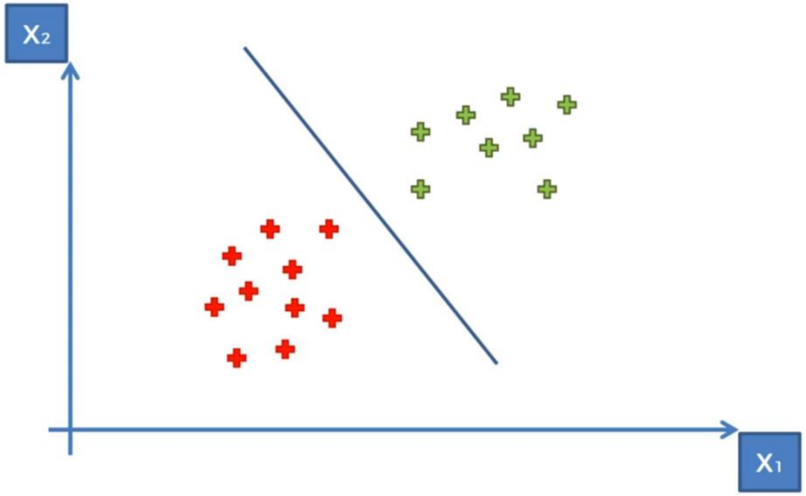
SVM. Part 2

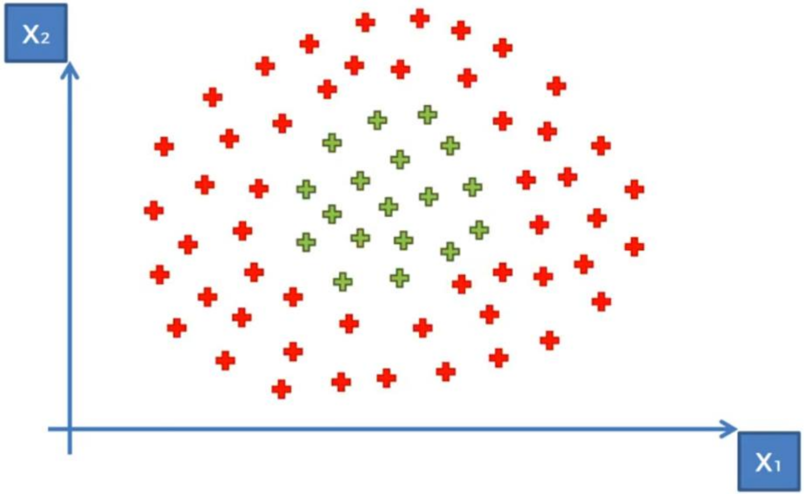


SVM. Part 2

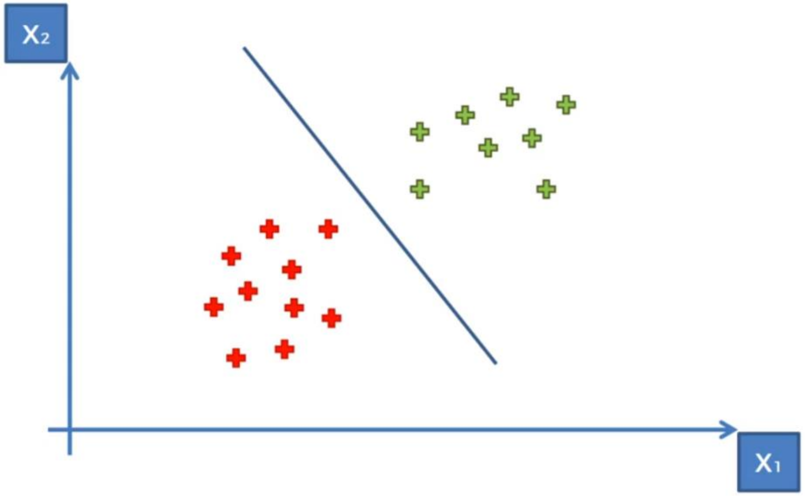

X2
X1
?
The data points in this case are not linearly separable

Linearly separable vs not separable

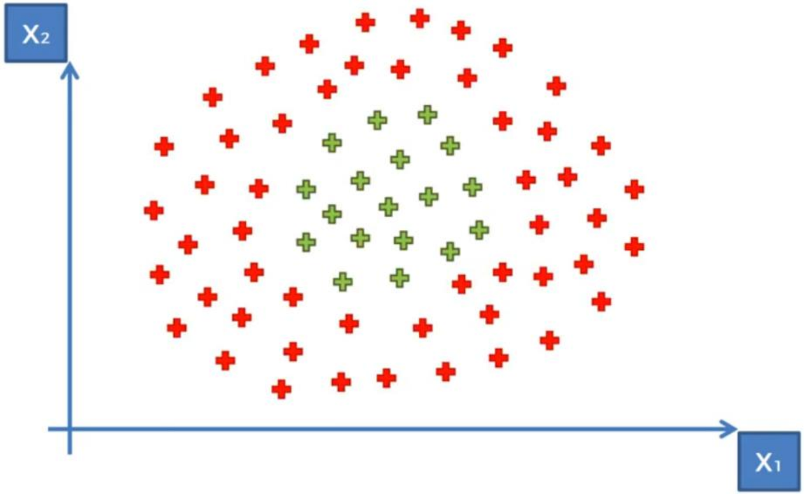




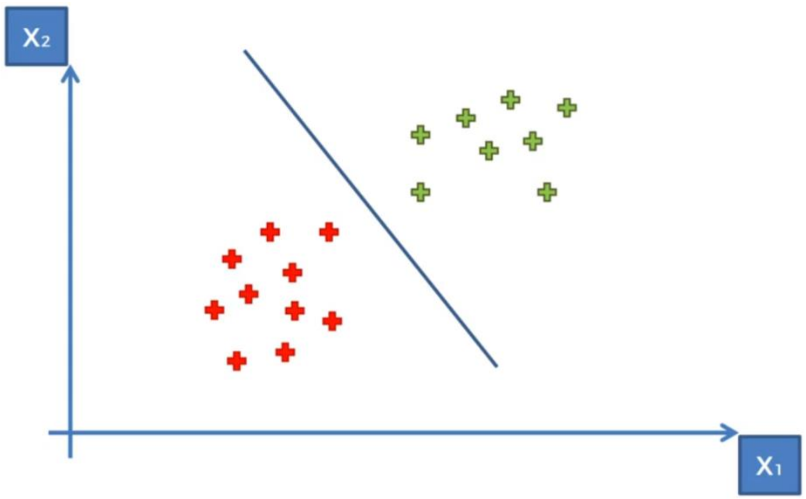
Linearly separable vs not separable



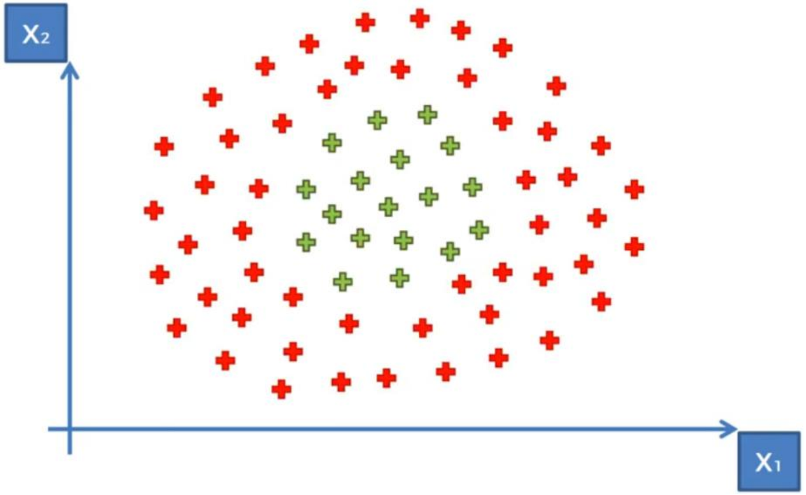
SVM has an assumption that it can be linearly separated



Linearly separable vs not separable

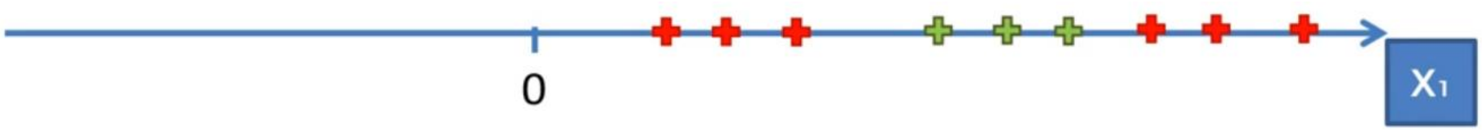


SVM has an assumption that it can be linearly separated.
How?
By adding extra dimension



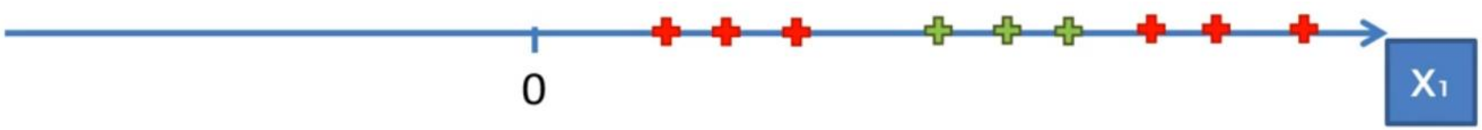
Mapping to a higher dimension

Let's start w/ the simple example – 1D



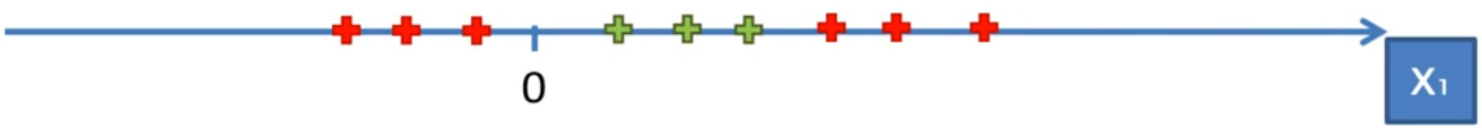
Mapping to a higher dimension

Let's apply a function
f = x - 5



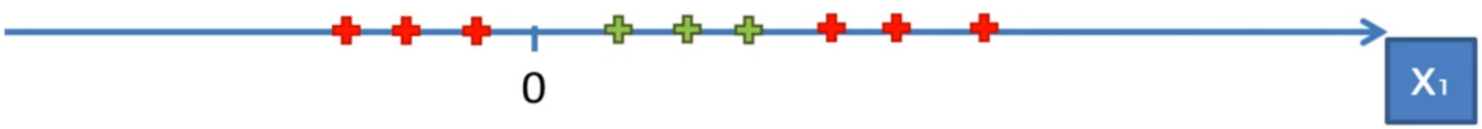
Mapping to a higher dimension

Let's apply a
function
$f = x - 5$

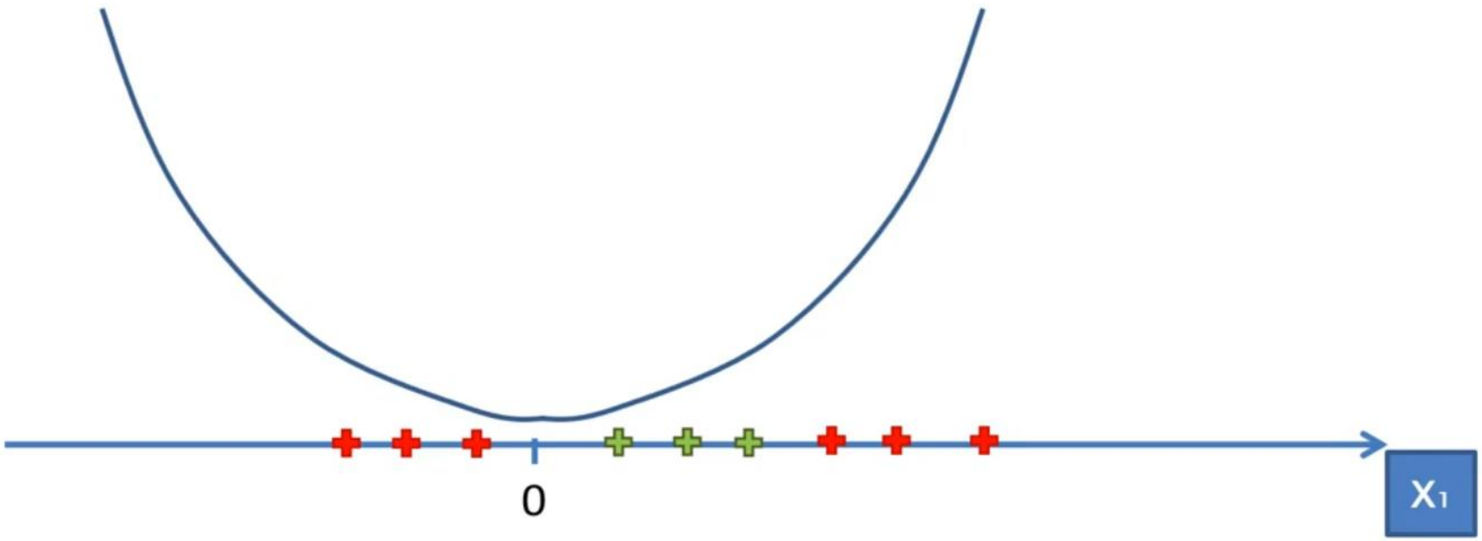


Mapping to a higher dimension

Now let's square all that:
f = (x-5)^2

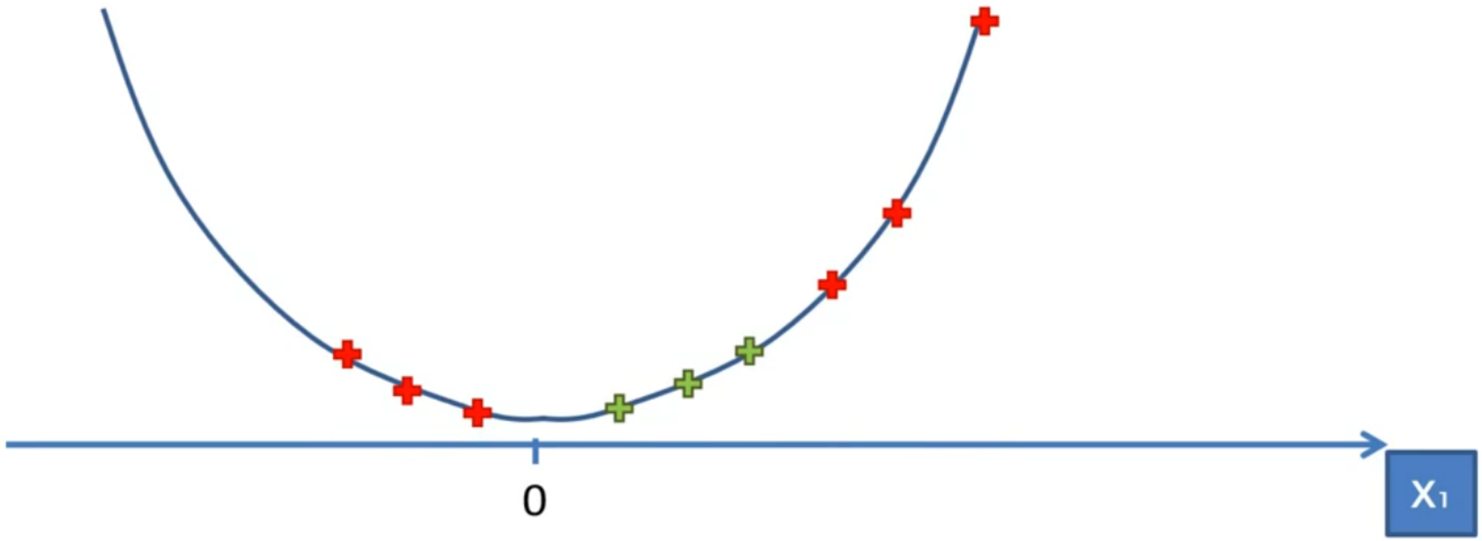


Mapping to a higher dimension



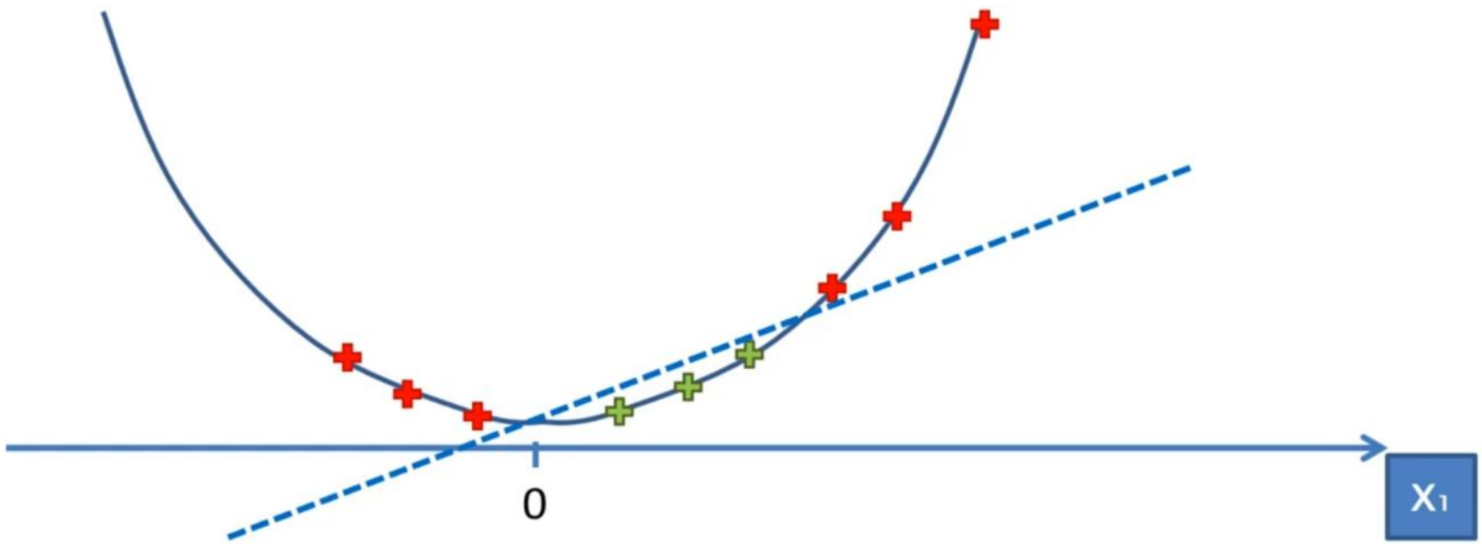
Now let's square all that:
f = (x-5)^2

Mapping to a higher dimension



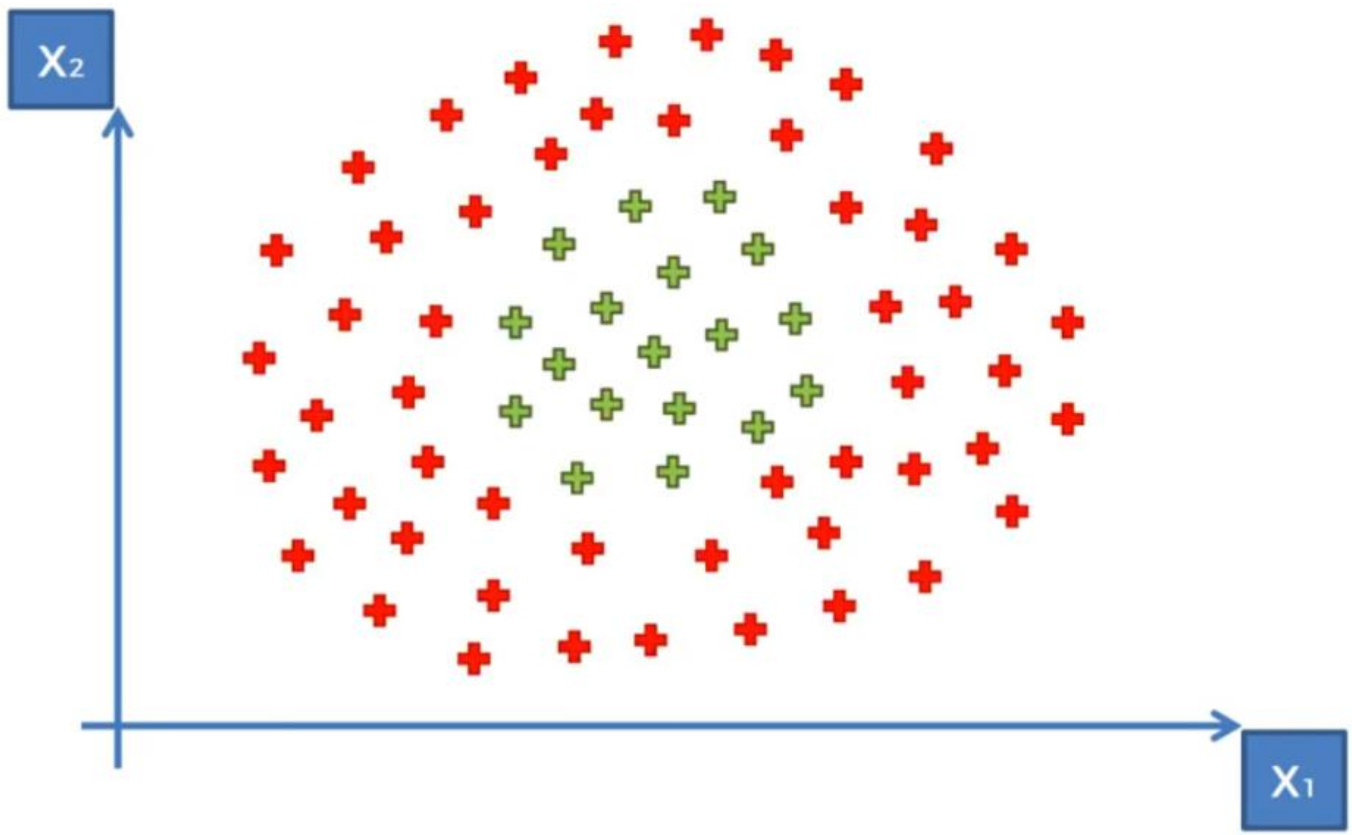
Now let's square all that:
f = (x-5)^2

Mapping to a higher dimension

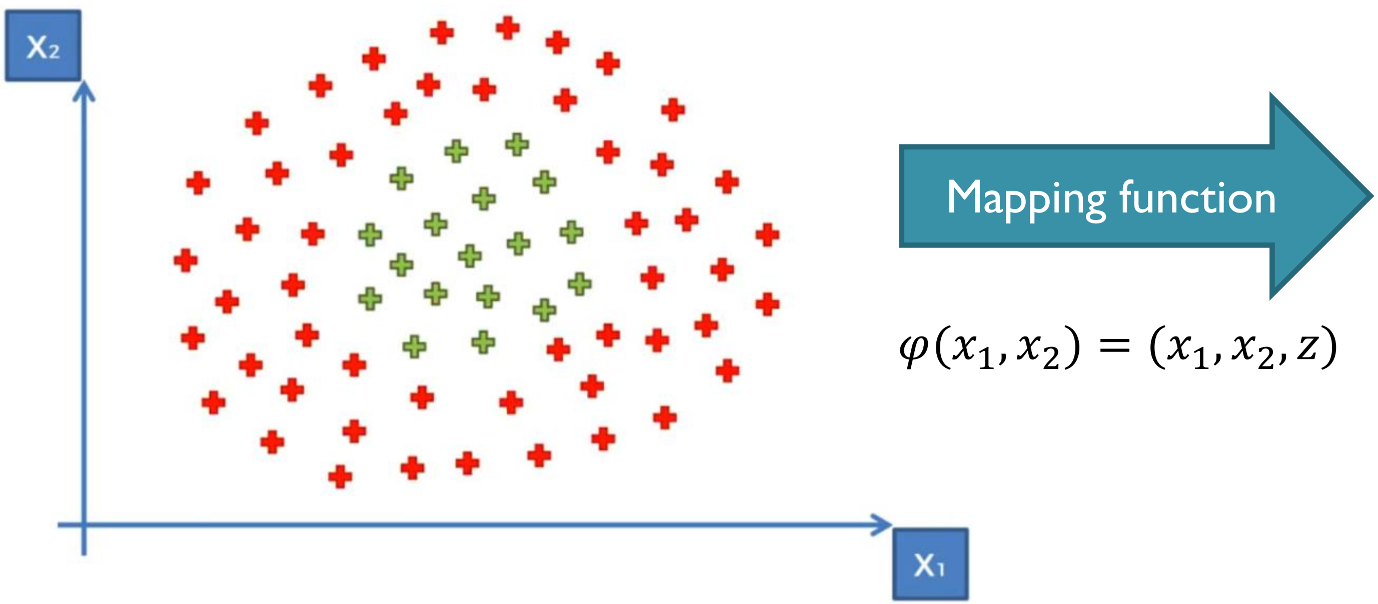


Finally we see that we can separate it

Mapping to a higher dimension

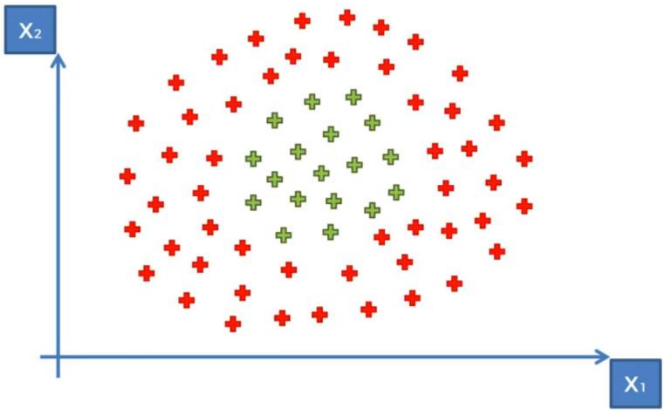


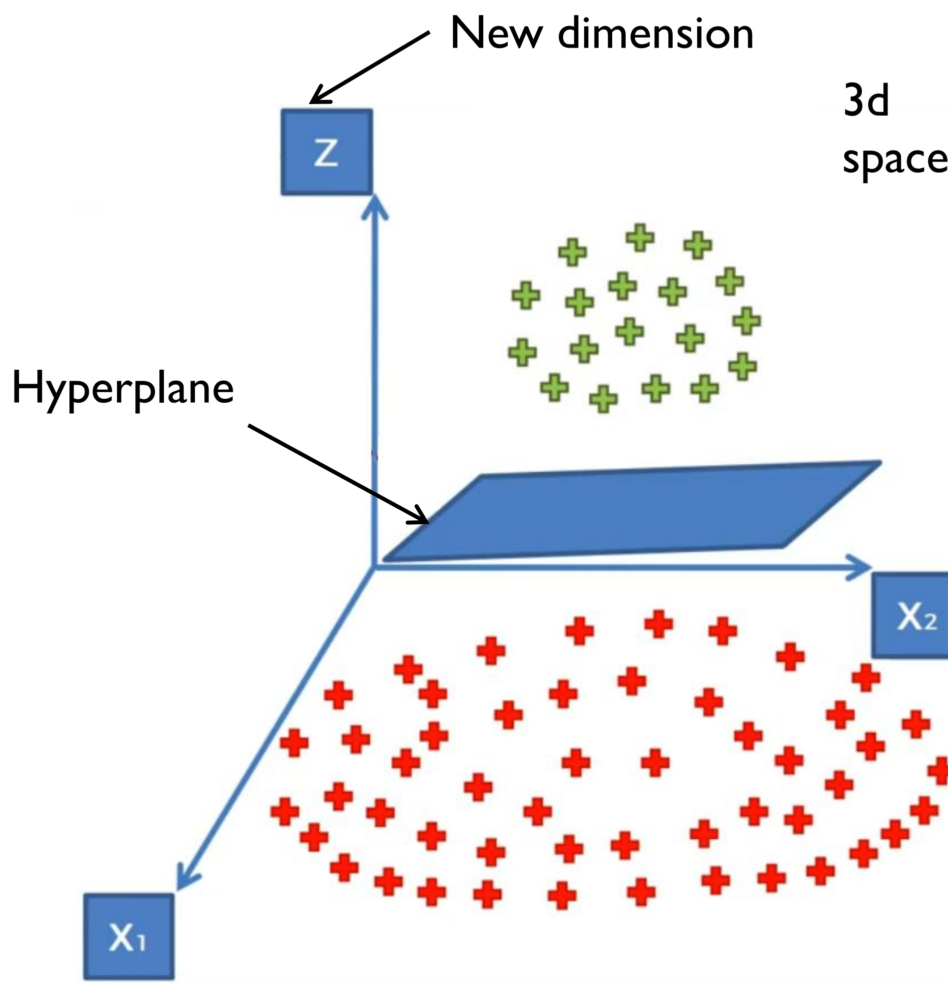
Mapping to a higher dimension



$$\varphi(x_1, x_2) = (x_1, x_2, z)$$

Mapping to a higher dimension

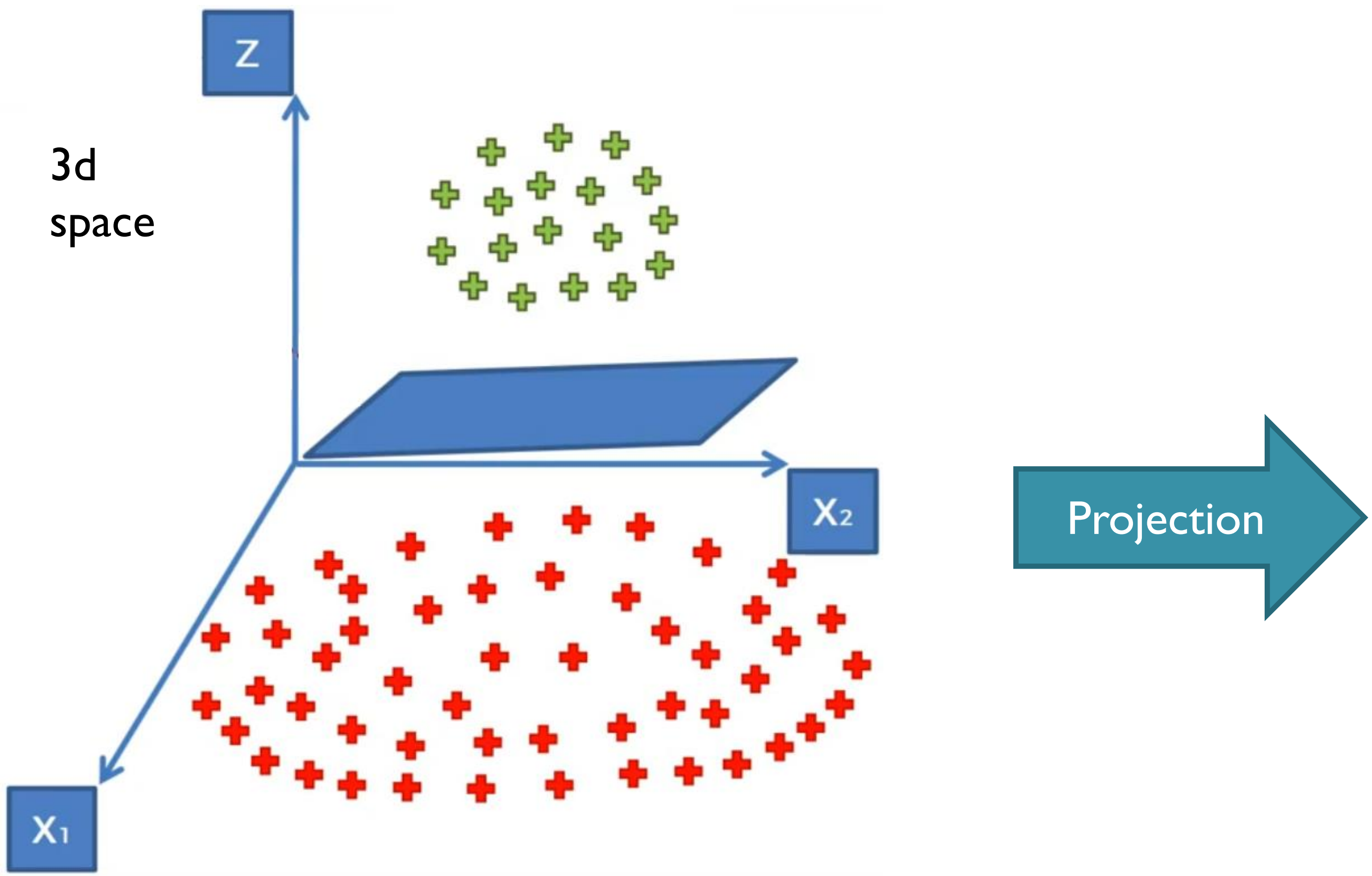




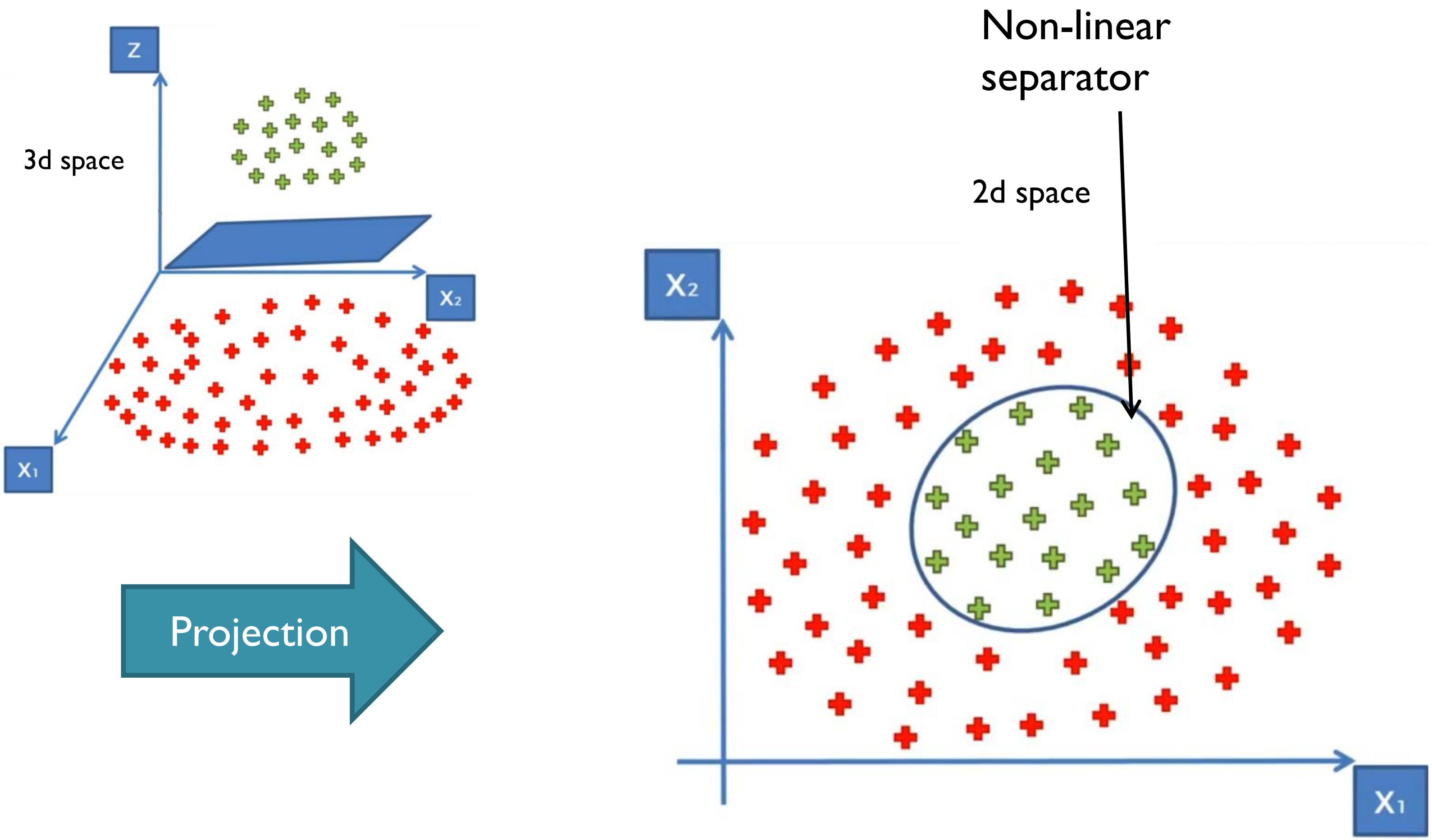
Mapping function

$$\varphi(x_1, x_2) = (x_1, x_2, z)$$

Mapping to a higher dimension

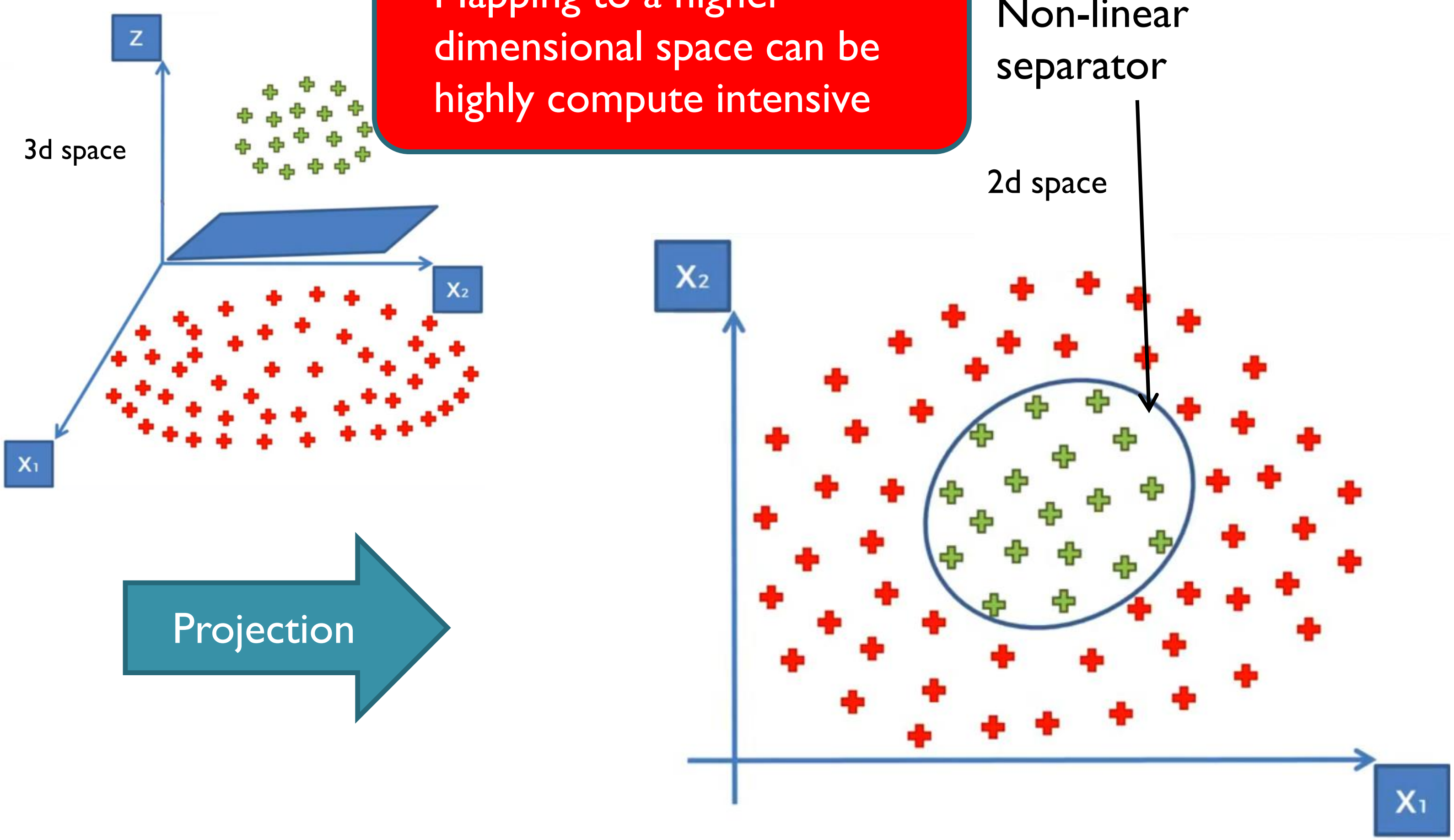


Mapping to a higher dimension



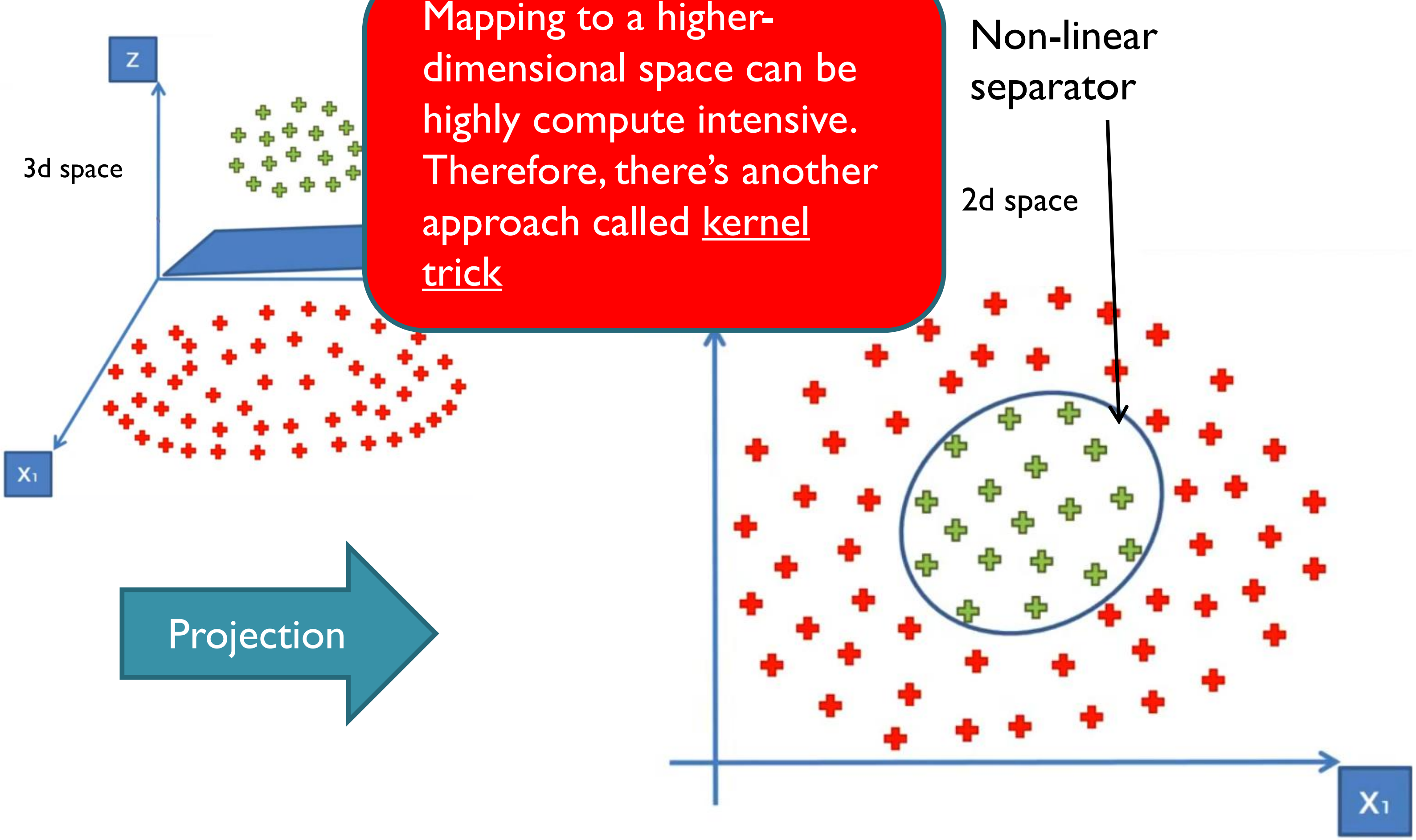
Mapping to a higher dimension

Mapping to a higher-dimensional space can be highly compute intensive



Mapping to a higher dimension

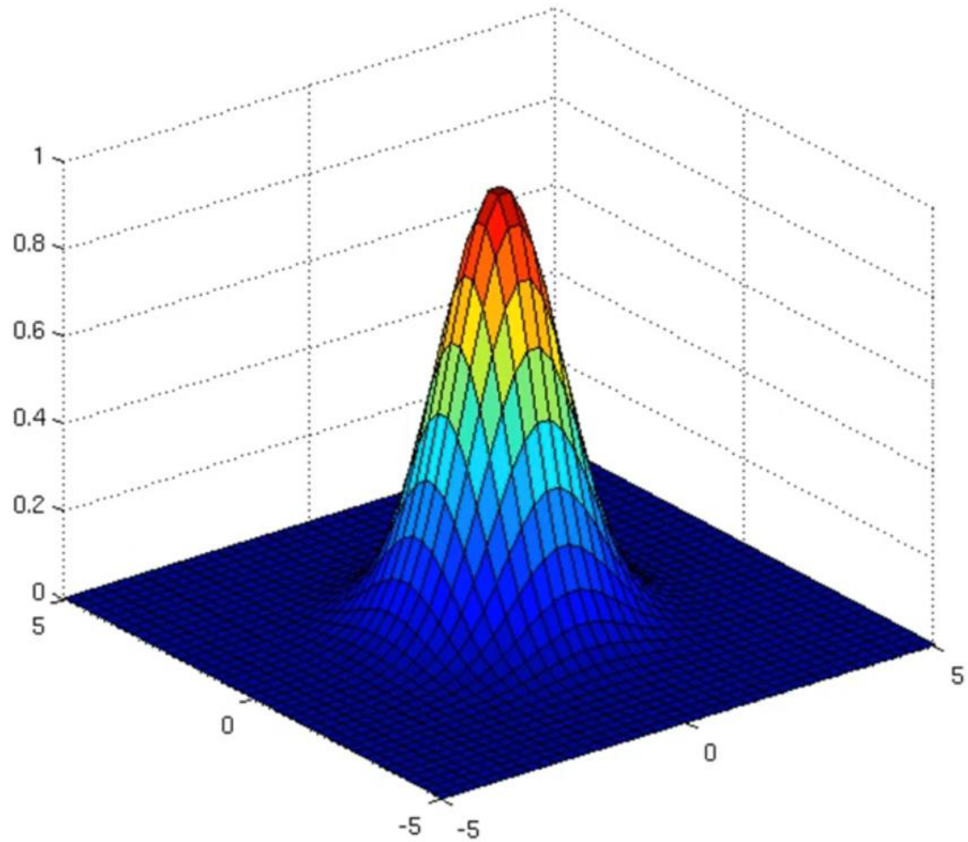
Mapping to a higher-dimensional space can be highly compute intensive. Therefore, there's another approach called kernel trick



Kernel trick

$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\left\|\vec{x}-\vec{l}^i\right\|^2}{2\sigma^2}}$$

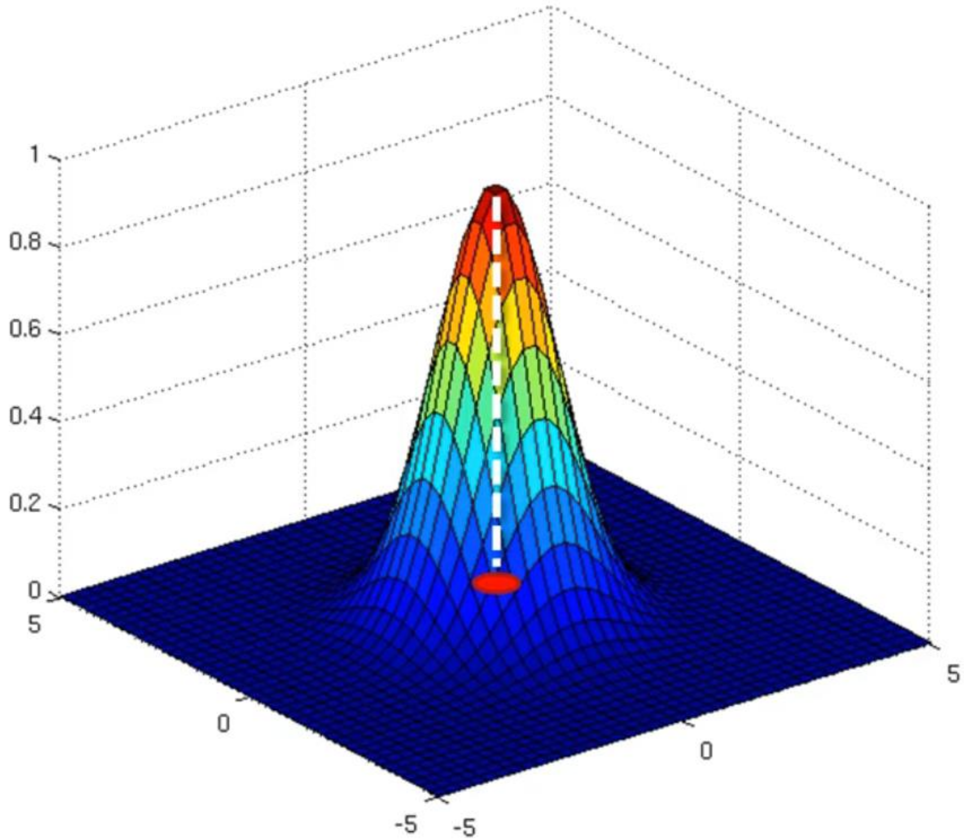
Kernel trick



Gaussian
Radial basis
function (RBF)

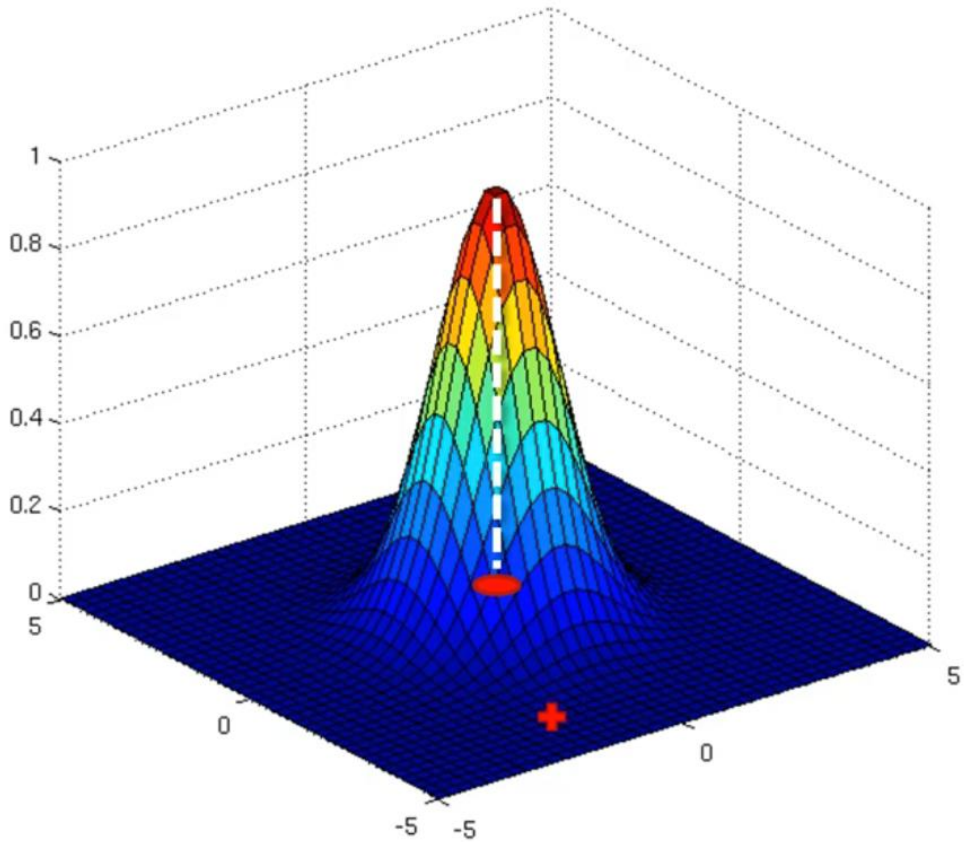
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$$

Kernel trick



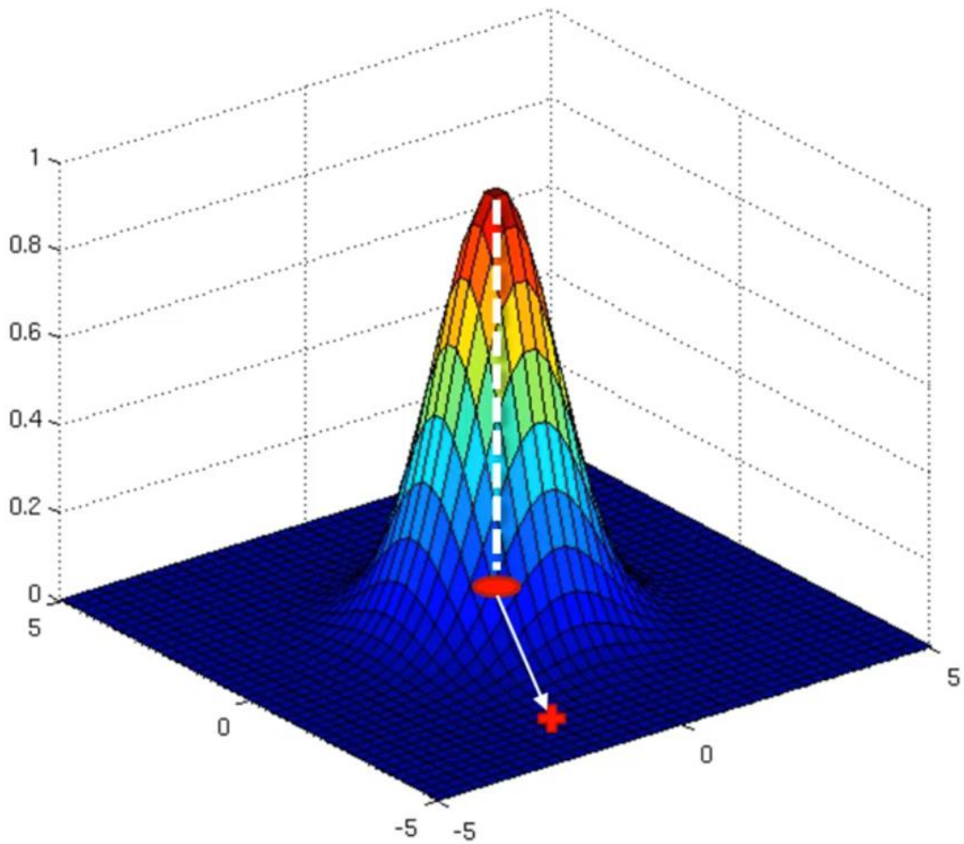
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\left\|\vec{x}-\vec{l}^i\right\|^2}{2\sigma^2}}$$

Kernel trick



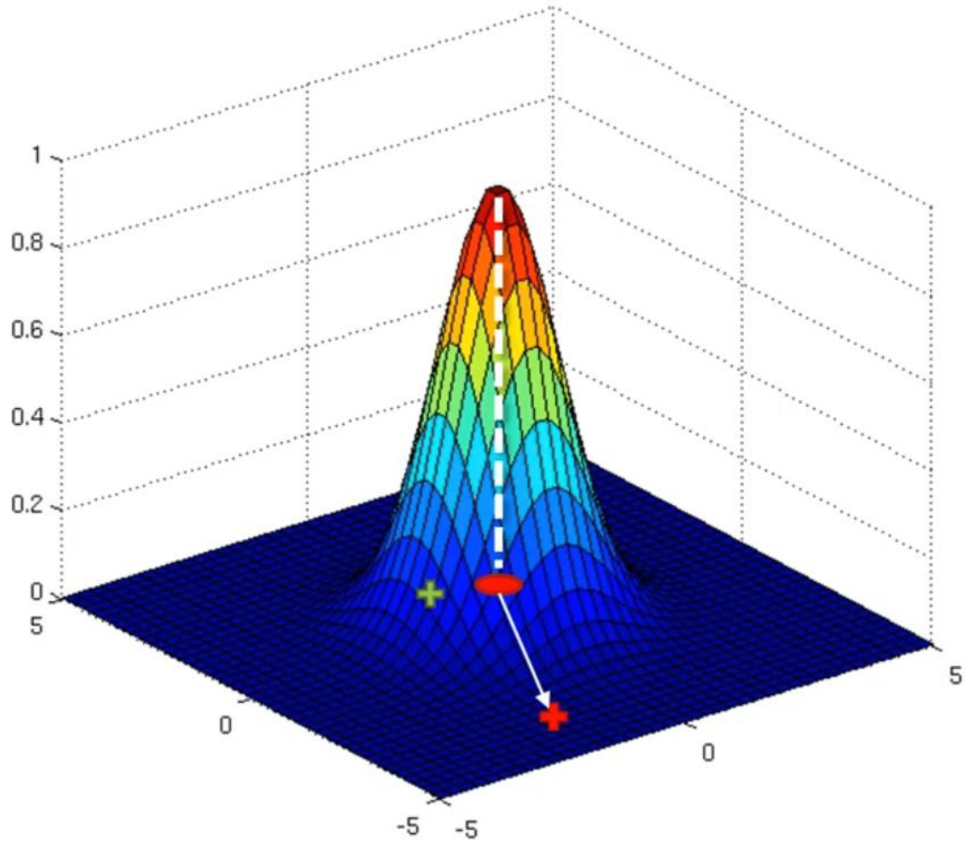
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x} - \vec{l}^i\|^2}{2\sigma^2}}$$

Kernel trick



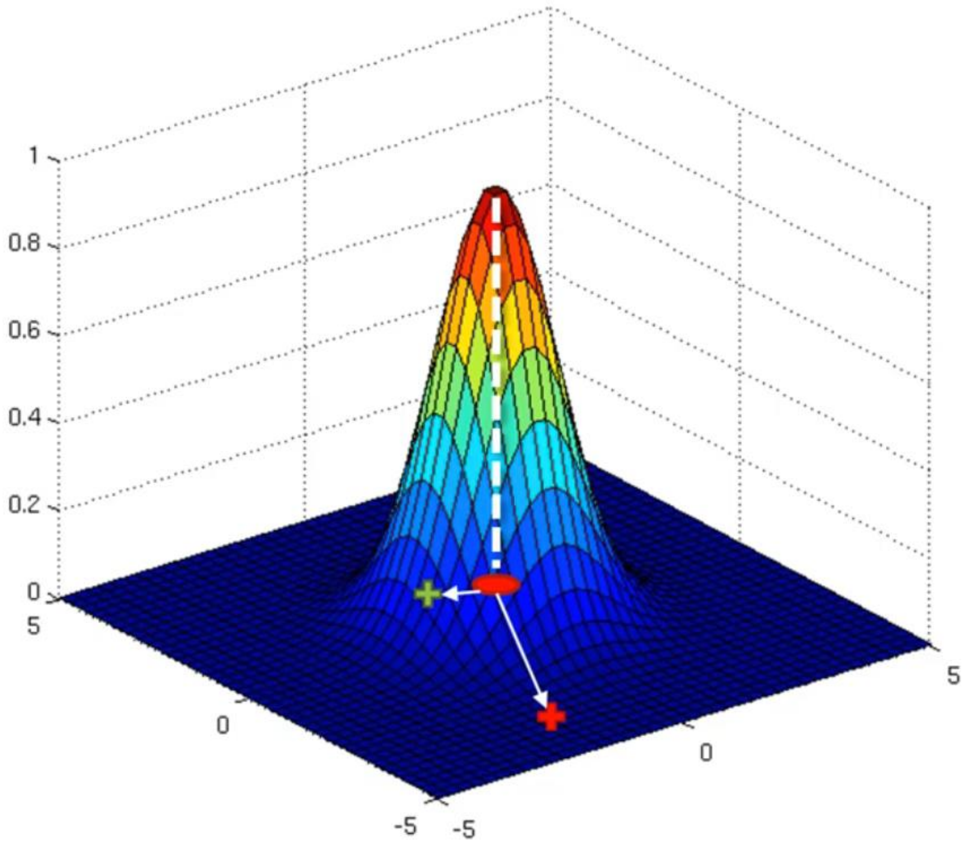
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\left\|\vec{x}-\vec{l}^i\right\|^2}{2\sigma^2}}$$

Kernel trick



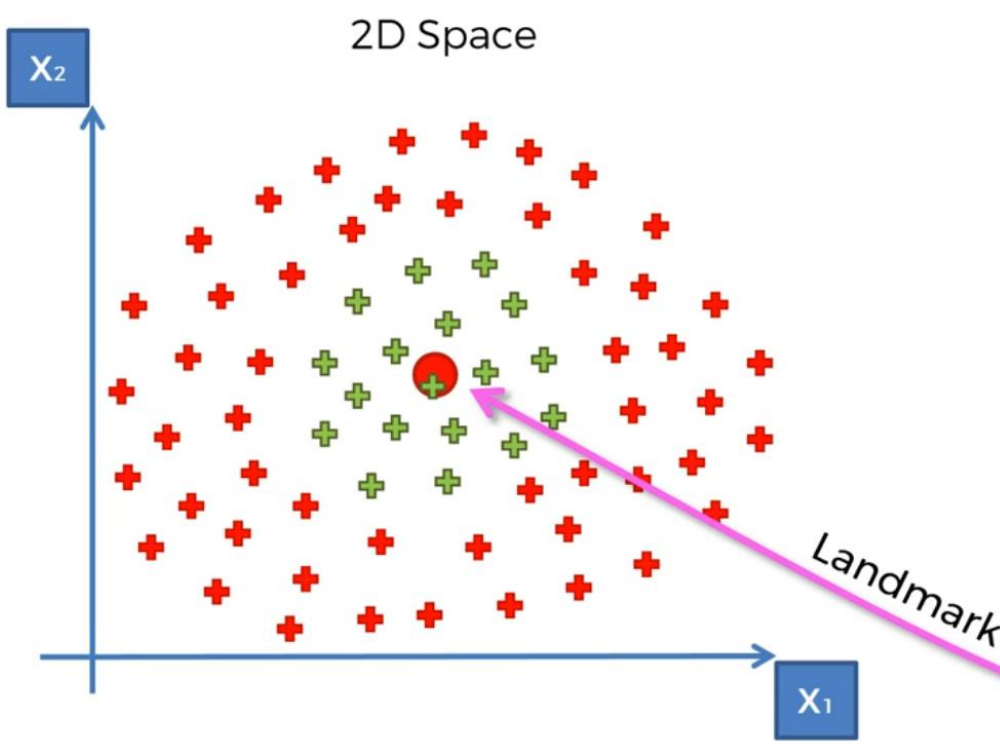
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\left\|\vec{x}-\vec{l}^i\right\|^2}{2\sigma^2}}$$

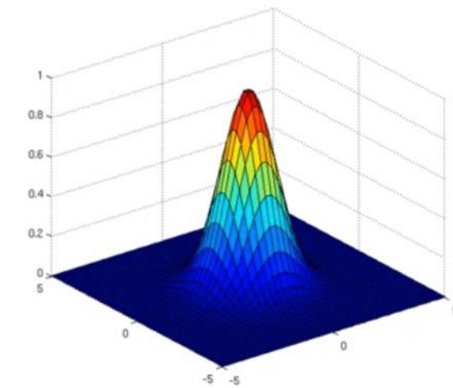
Kernel trick



$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x} - \vec{l}^i\|^2}{2\sigma^2}}$$

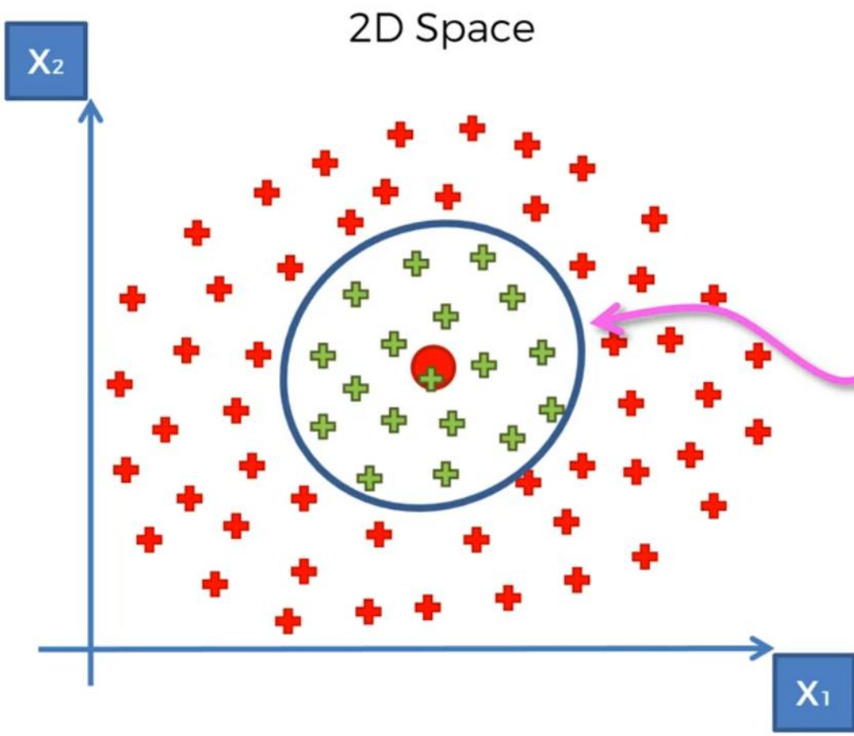
Kernel trick

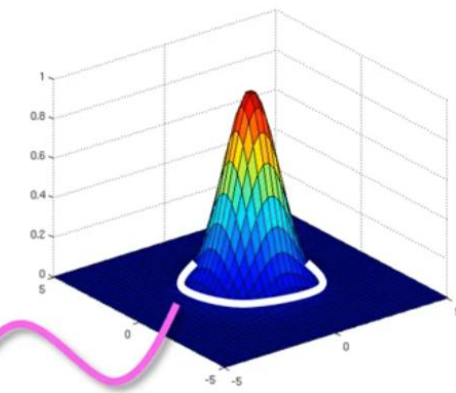




$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

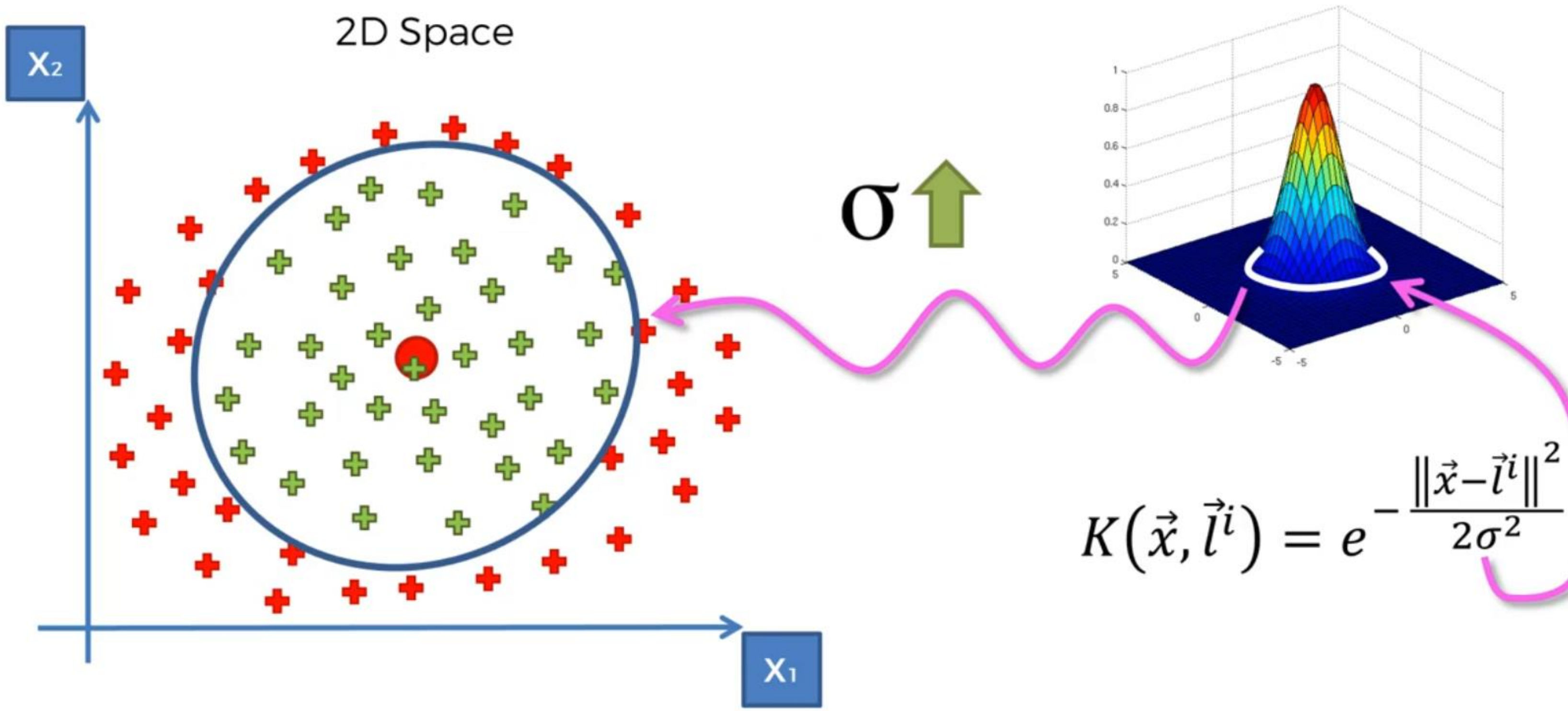
Kernel trick





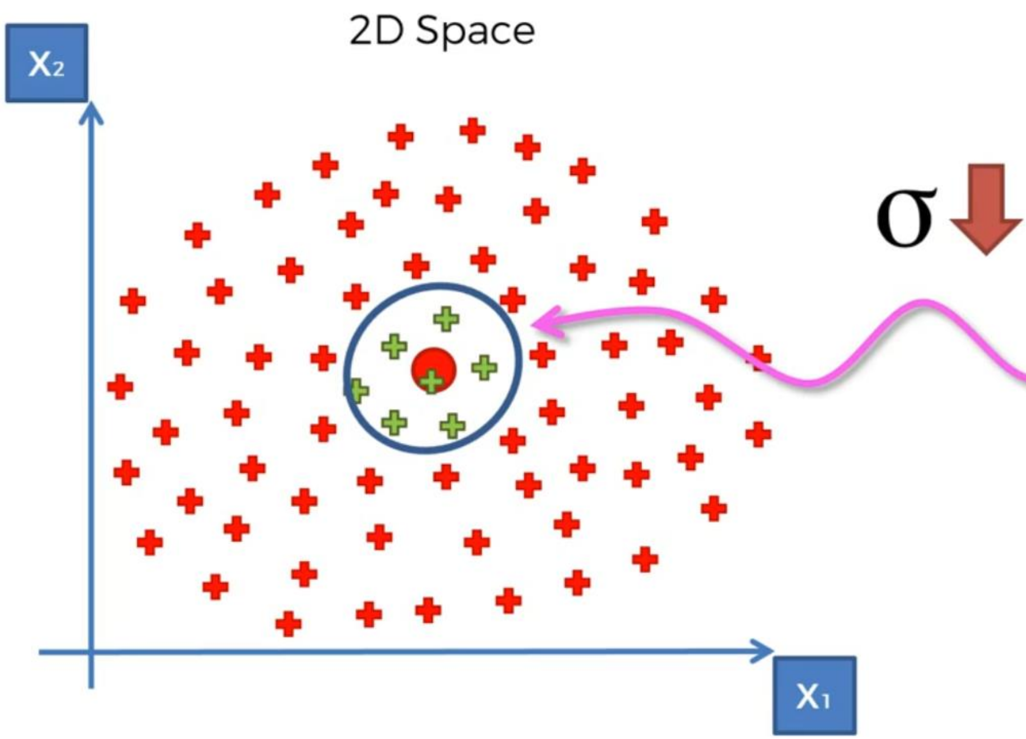
$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$$

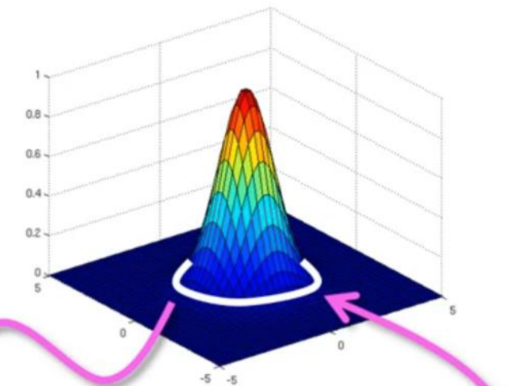
Kernel trick



$$K(\vec{x}, \vec{l}^{i}) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

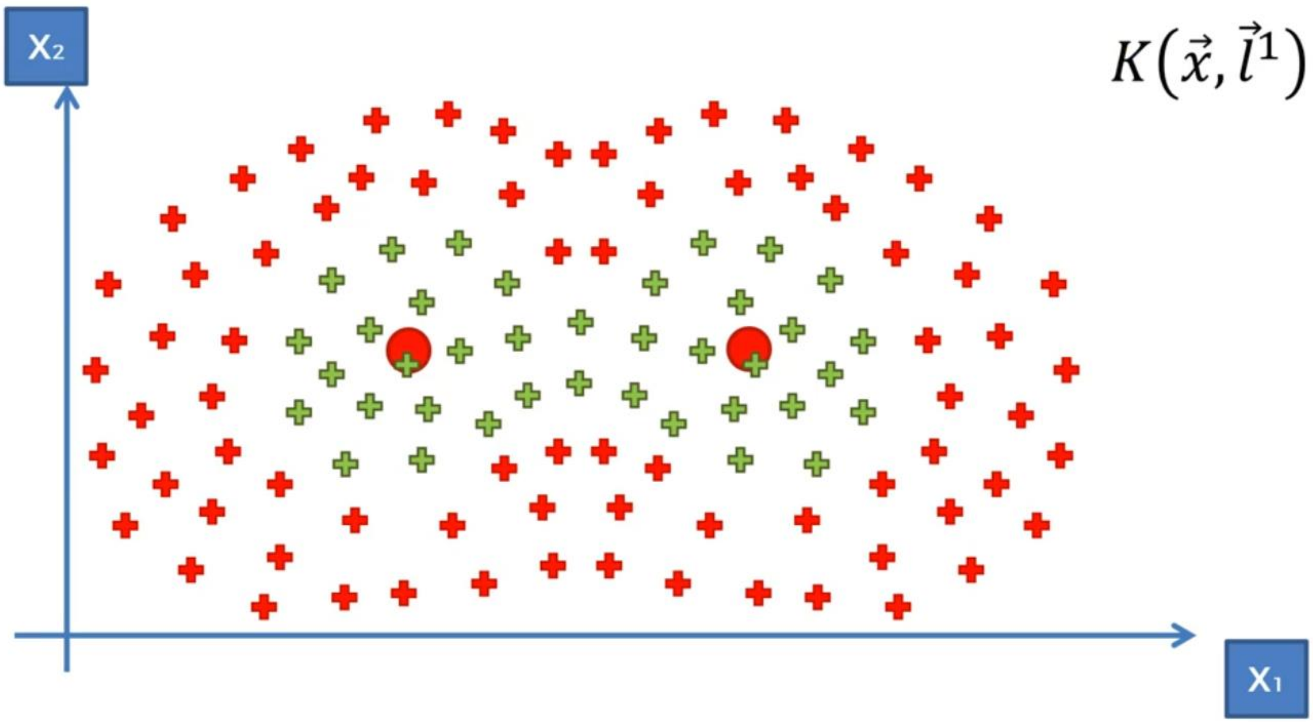
Kernel trick





$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$$

Kernel trick



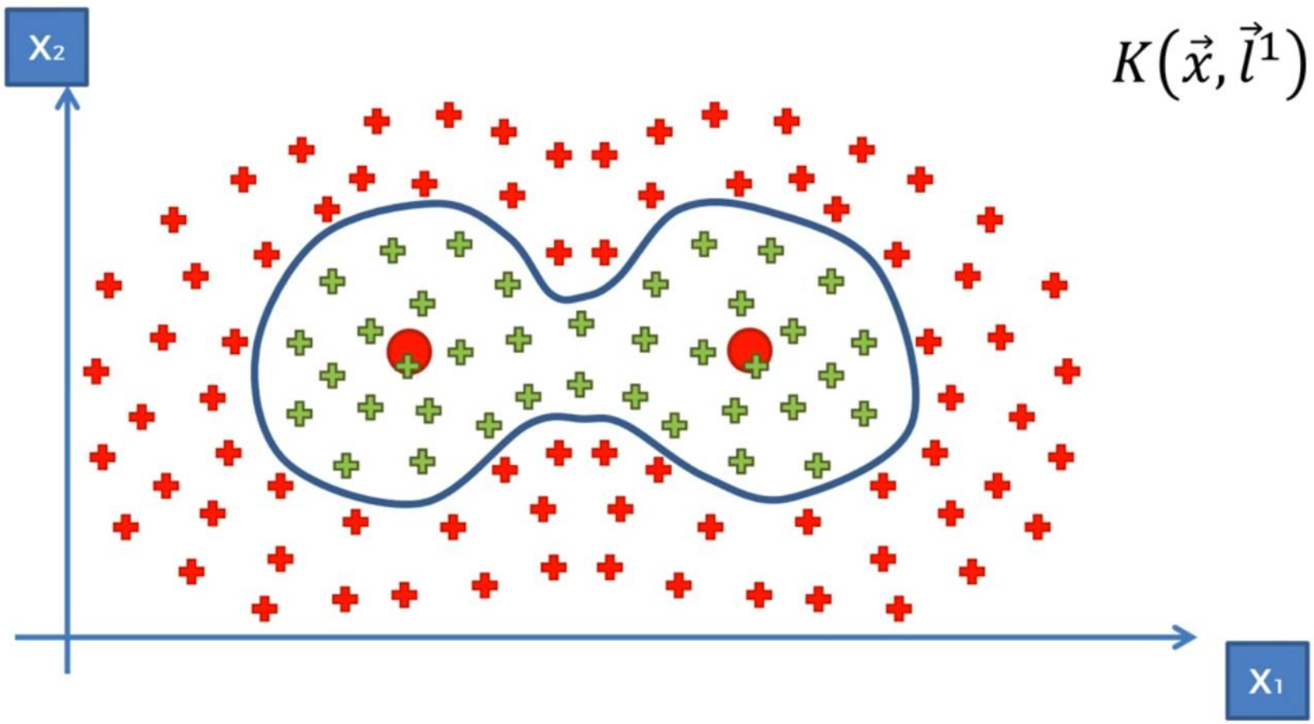
$$K(\vec{x}, \vec{l}^{1}) + K(\vec{x}, \vec{l}^{2})$$

(Simplified Formula)

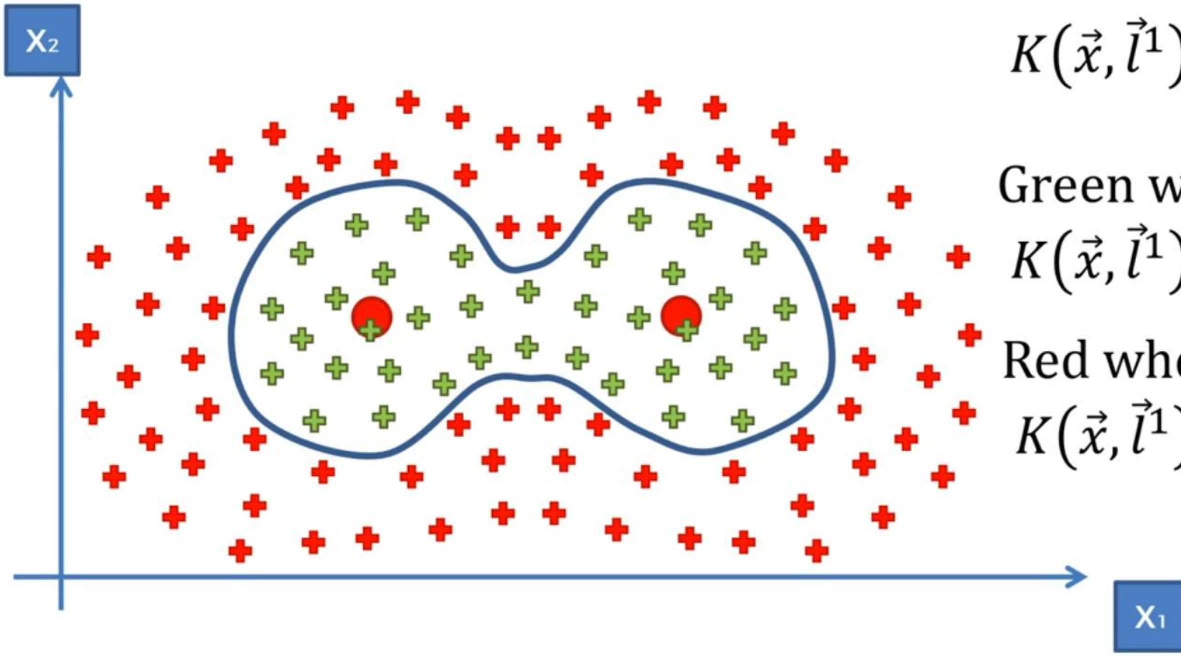
Kernel trick

$$K(\vec{x},\vec{l}^1)+K(\vec{x},\vec{l}^2)$$

(Simplified Formula)



Kernel trick



$$K(\vec{x},\vec{l}^1)+K(\vec{x},\vec{l}^2)$$

(Simplified Formula)

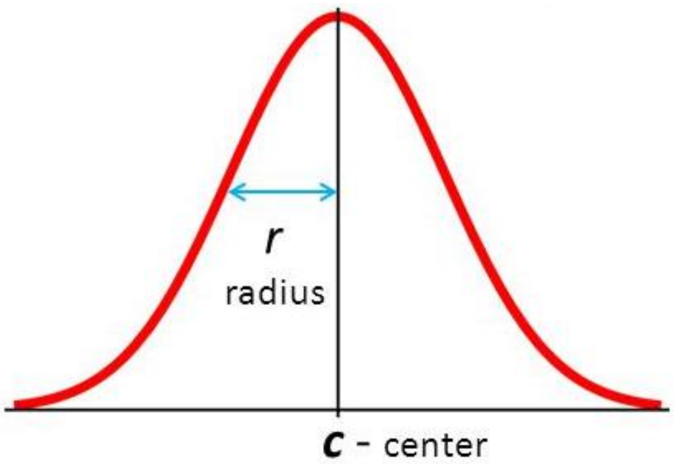
Green when:

$$K(\vec{x},\vec{l}^1)+K(\vec{x},\vec{l}^2)>0$$

Red when:

$$K(\vec{x},\vec{l}^1)+K(\vec{x},\vec{l}^2)=0$$

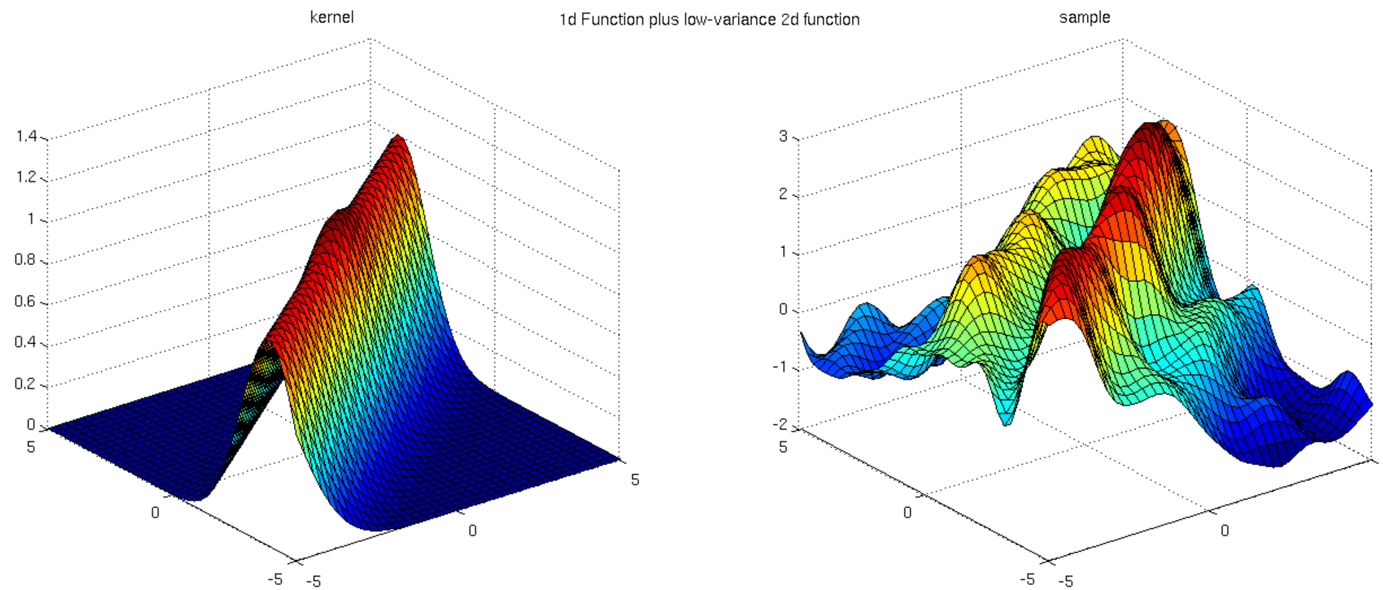
Types of kernel functions



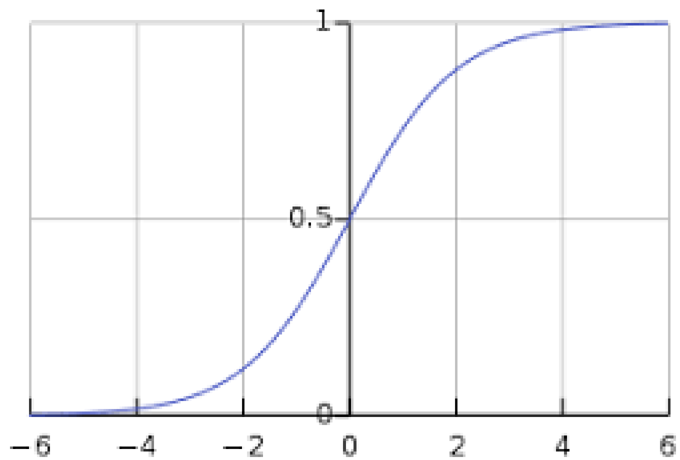
$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

Gaussian
Radial basis
function (RBF)

Types of kernel functions



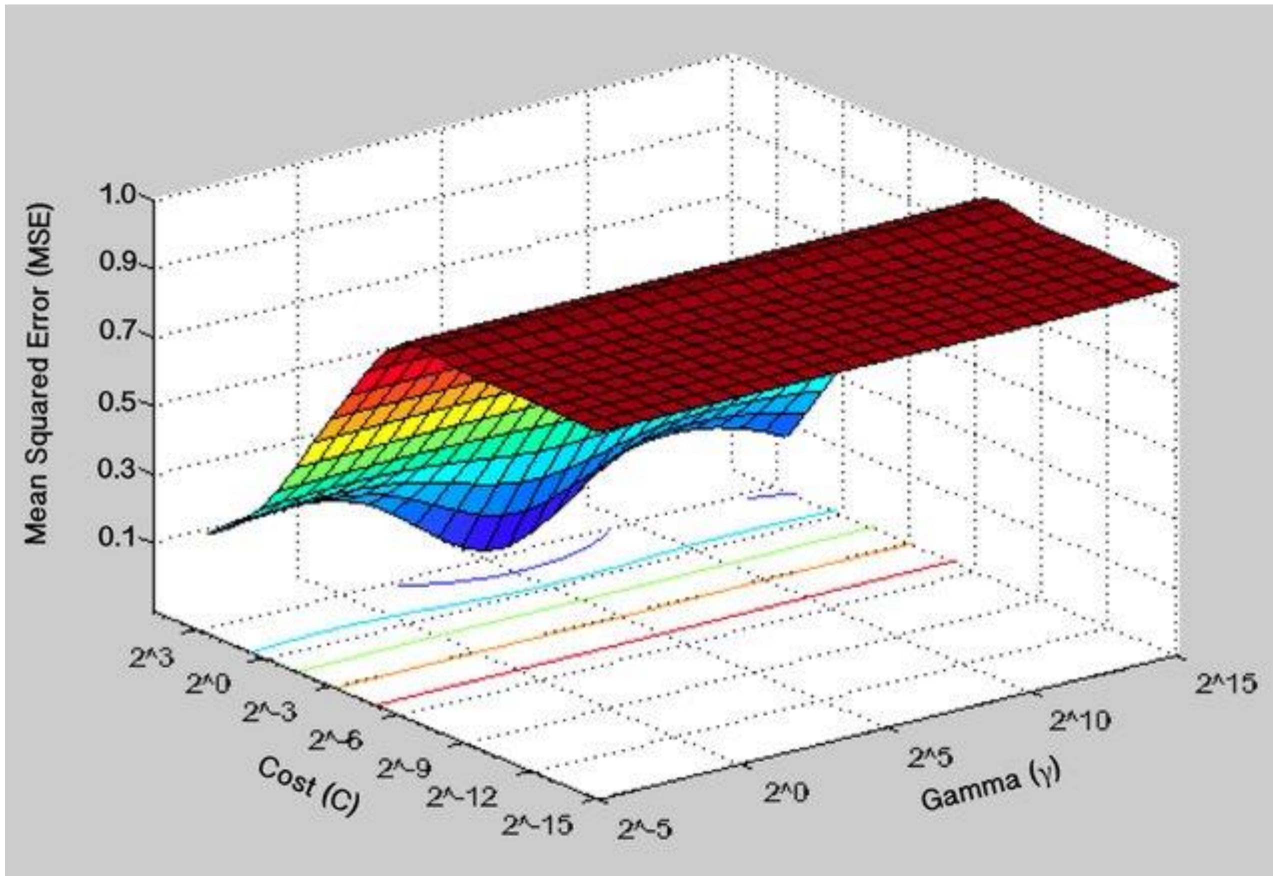
Types of kernel functions



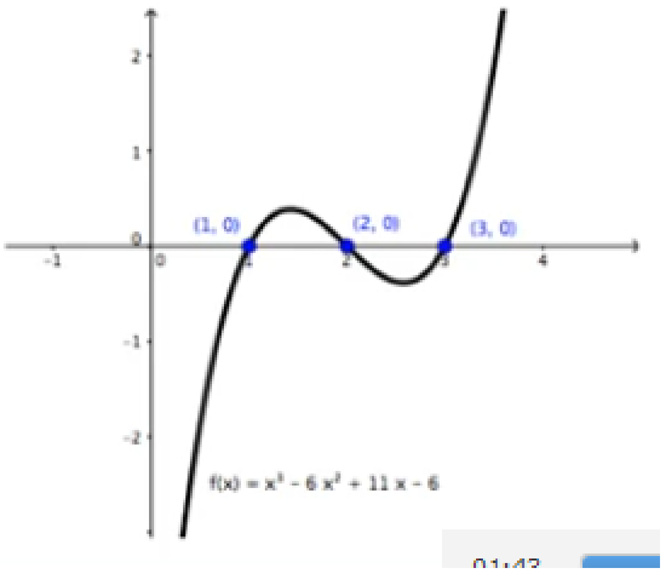
$$K(X, Y) = \tanh(\gamma \cdot X^T Y + r)$$

Sigmoid kernel

Types of kernel functions



Types of kernel functions



$$K(X,Y) = (\gamma \cdot X^T Y + r)^d, \gamma > 0$$

Polynomial kernel

Additional readings

- https://mlkernels.readthedocs.io/en/latest/index.html
- https://en.wikipedia.org/wiki/Radial_basis_function

This is it ;-)