KERNEL TRICKS

HOSSEIN JAVIDNIA

POSTDOCTORAL RESEARCH FELLOW,

ADAPT CENTRE, TRINITY COLLEGE DUBLIN

INTRODUCTION

1

Support vector classification relies on the notion of linearly separable data.

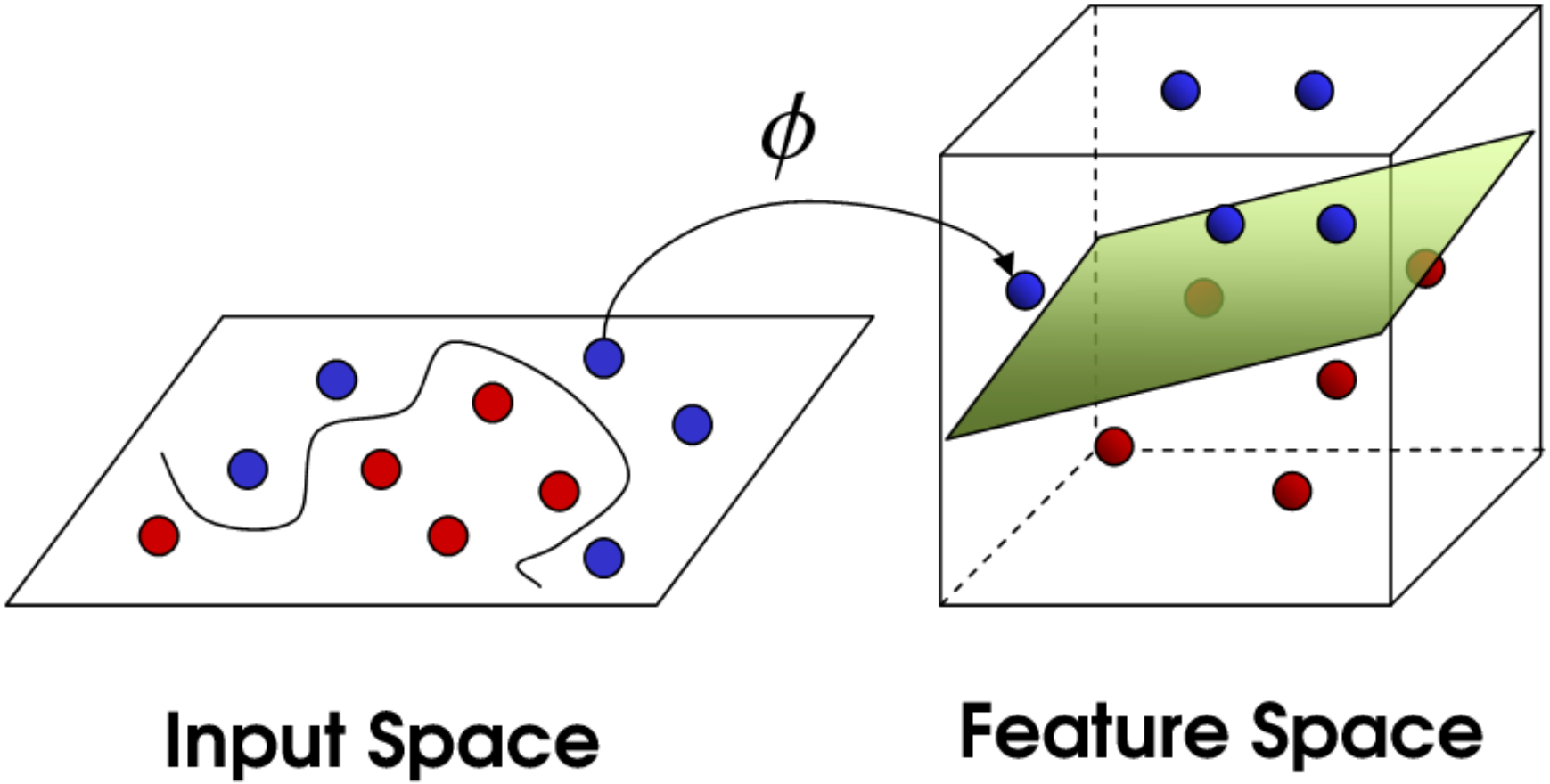
2

In practice data is often very far from being linearly separable, and we need to transform the data into a higher dimensional space in order to fit a support vector classifier.

3

In machine learning, a "kernel" is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem.

FEATURE TRANSFORMATION



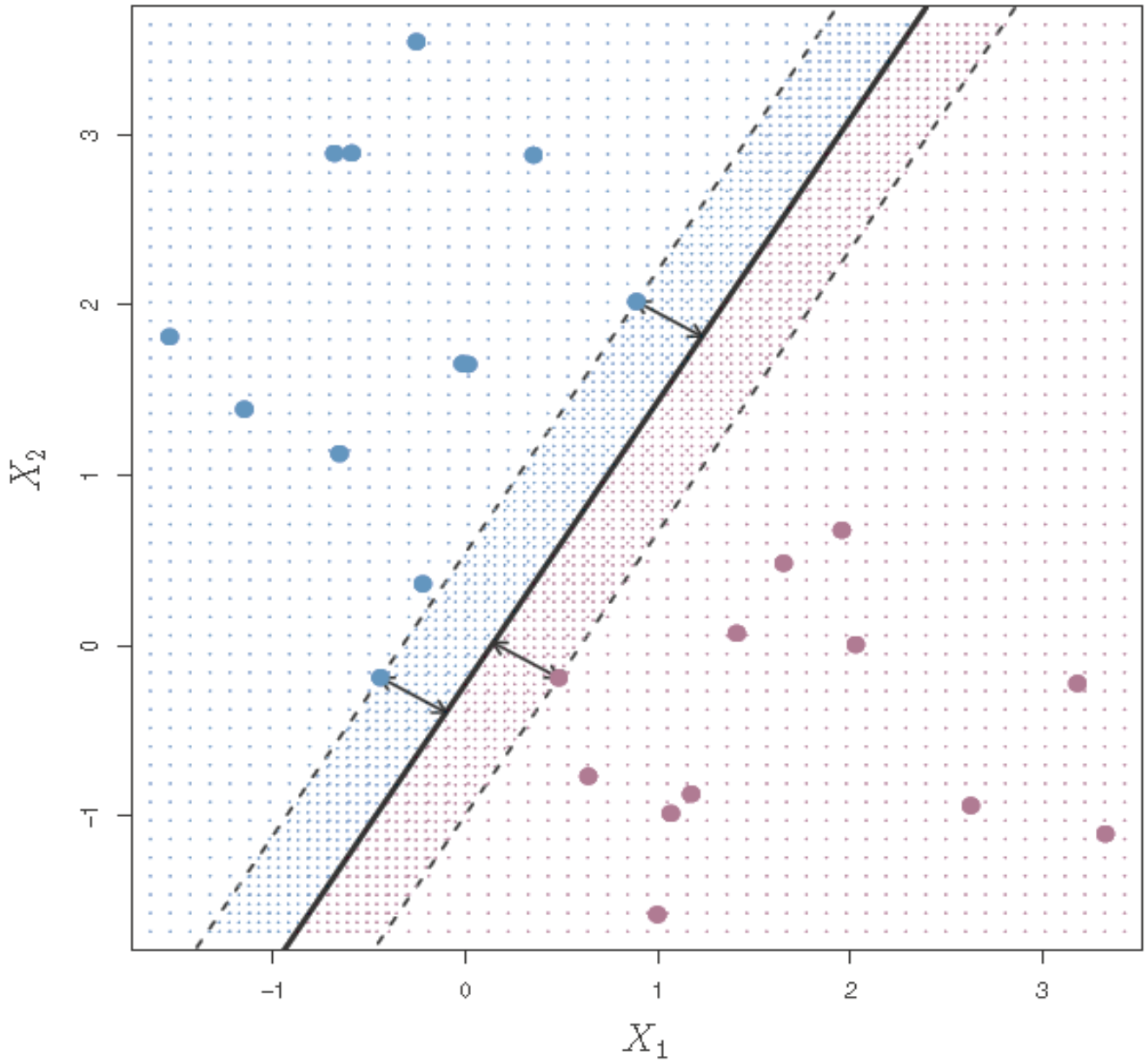
FLASHBACK - SVM

Training a linear support vector classifier, like nearly every problem in machine learning, is an optimization problem.

We maximize the *margin* - the distance separating the closest pair of data points belonging to opposite classes.

These points are called the support vectors, because they are the data observations that "support", or determine, the decision boundary.

To train a support vector classifier, we find the *maximal margin hyperplane*, or *optimal separating hyperplane*, which optimally separates the two classes in order to generalize to new data and make accurate classification predictions.


X_2
3
2
1
0
−1
−1
0
1
2
3
X_1

NON-LINEAR TRANSFORMATIONS

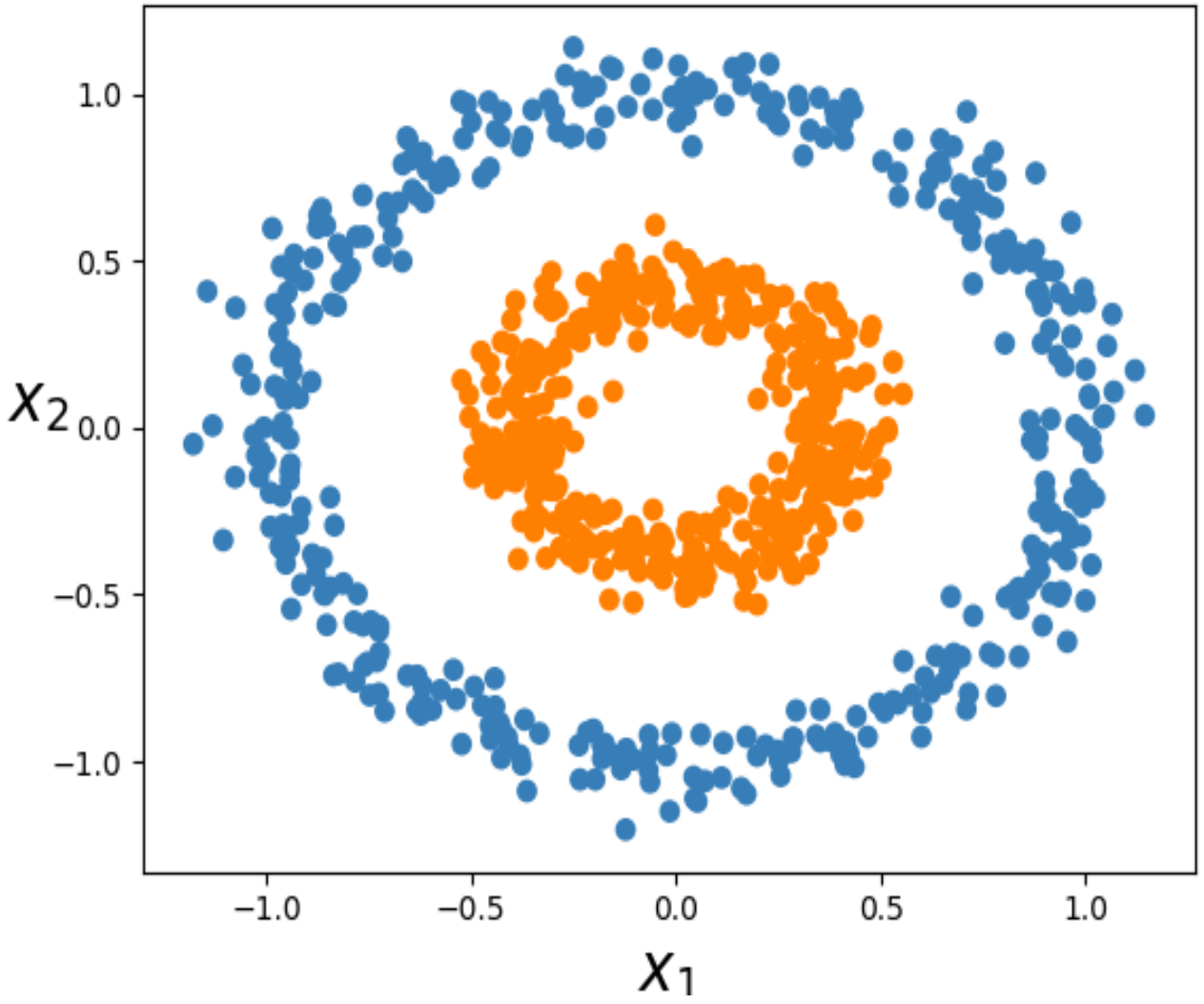
- If the data is not linearly separable in the original, or input, space then we apply transformations to the data, which MAP the data from the original space into a higher dimensional feature space.

- The goal is that after the transformation to the higher dimensional space, the classes are now linearly separable *in this higher dimensional feature space*. We can then fit a decision boundary to separate the classes and make predictions.

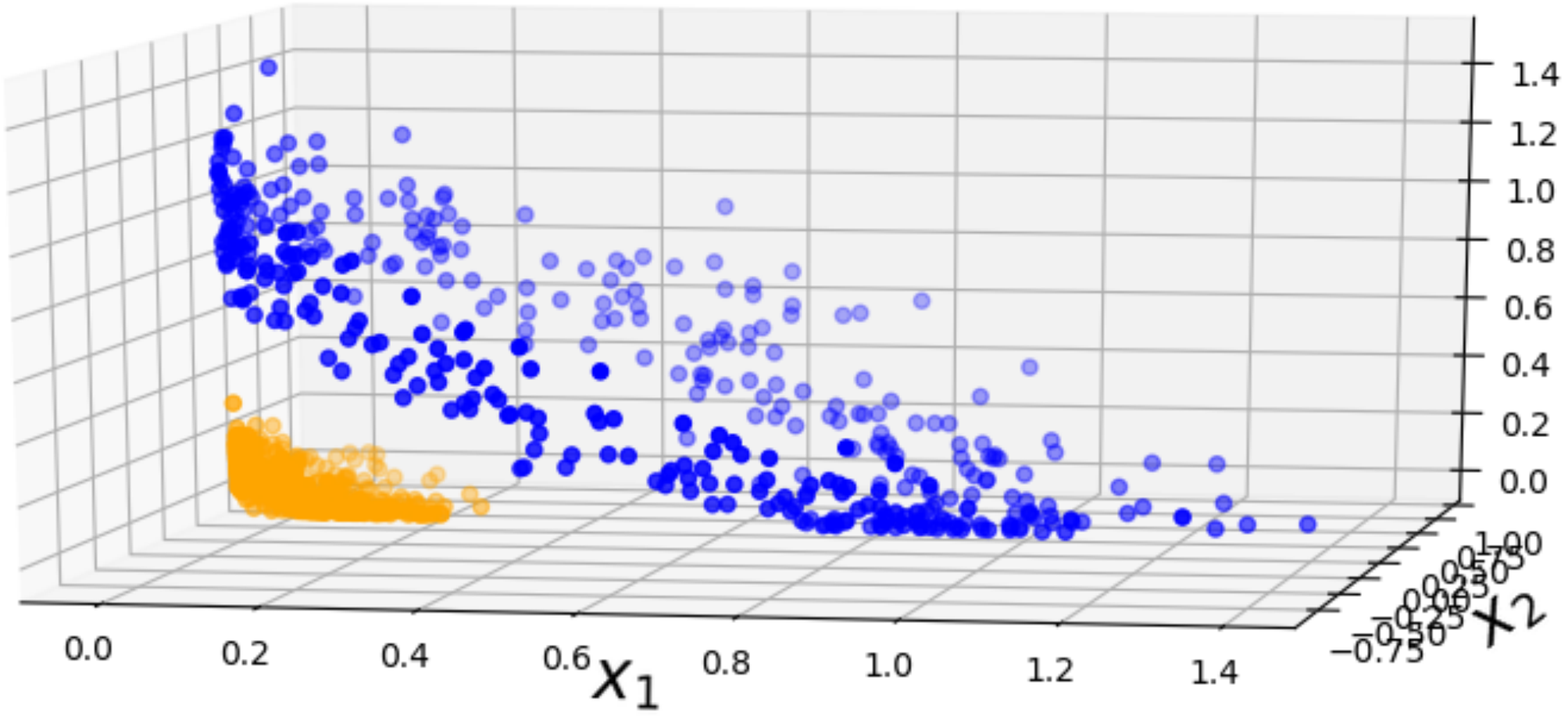
EXAMPLE – 2D TO 3D

This data is not linearly separable. So, we use Second degree polynomial mapping, as follows to map the data from 2D to 3D:

$$\phi(\mathbf{X}) = \phi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} x_1^2 \\ \sqrt{x_1 x_2} \\ x_2^2 \end{pmatrix}$$



EXAMPLE – 2D TO 3D



CHALLENGES & SOLUTION

Challenge:

- In real applications, there might be many features in the data and applying transformations that involve many polynomial combinations of these features will lead to extremely high and impractical computational costs.

Solution:

- The **kernel trick** provides an elegant solution to this problem.

KERNEL DEFINITION

- A function that takes as its input, the feature vectors in original space and returns the dot product of the transformed vectors in feature space, is called KERNEL FUNCTION.

- If we have data $\mathbf{x}, \mathbf{z} \in X$ and a map $\phi: X \rightarrow \Re^n$ then:

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$$

BENEFIT

The ultimate benefit of the kernel trick is that the objective function we are optimizing to fit the higher dimensional decision boundary only includes the dot product of the transformed feature vectors. Therefore, we can just substitute these dot product terms with the kernel function, and we don't even use $\phi(\mathbf{x})$.

EXAMPLE

$$x = (x_1, x_2, x_3)$$
$$z = (z_1, z_2, z_3)$$
$$\phi(x) = (x_1x_1, x_1x_2, x_1x_3, x_2x_1, x_2x_2, x_2x_3, x_3x_1, x_3x_2, x_3x_3)$$
$$\phi(z) = (z_1z_1, z_1z_2, z_1z_3, z_2z_1, z_2z_2, z_2z_3, z_3z_1, z_3z_2, z_3z_3)$$

- Let's plug in some numbers to make this more intuitive: suppose

$$x = (1, 2, 3); z = (4, 5, 6).$$

Then:

$$\phi(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)$$
$$\phi(z) = (16, 20, 24, 20, 25, 30, 24, 30, 36)$$
$$\langle \phi(x), \phi(z) \rangle = 16 + 40 + 72 + 40 + 100 + 180 + 72 + 180 + 324 = 1024$$

- A lot of algebra, mainly because ϕ is a mapping from 3-dimensional to 9-dimensional space.

EXAMPLE

- Now let us use the kernel instead. Lets assume the kernel is $K(x, z) = (\langle x, z \rangle)^2$:

$$K(x, z) = (4 + 10 + 18)^2 = 32^2 = 1024$$