CS395: Selected CS1 (Introduction to Machine Learning)

Associate Prof. Wessam El-Behaidy

Fall 2021

References:

https://www.coursera.org/learn/machine-learning (Andrew Ng)

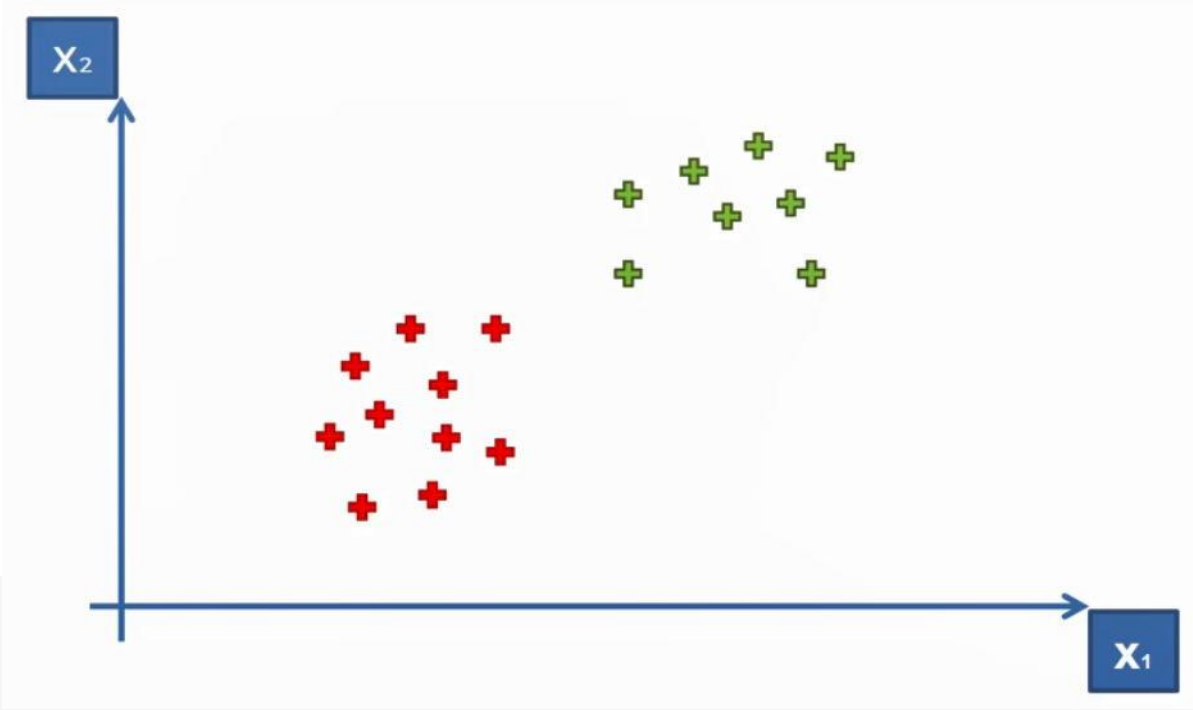
Machine learning A to Z: **Kirill Eremenko ©superdatascience**

Support Vector Machine

- Support Vector Machine (SVM) is a **supervised machine learning algorithm** that can be employed for both classification and regression purposes as support vector classification (SVC) and support vector regression (SVR).
- They are more commonly used in classification problems.

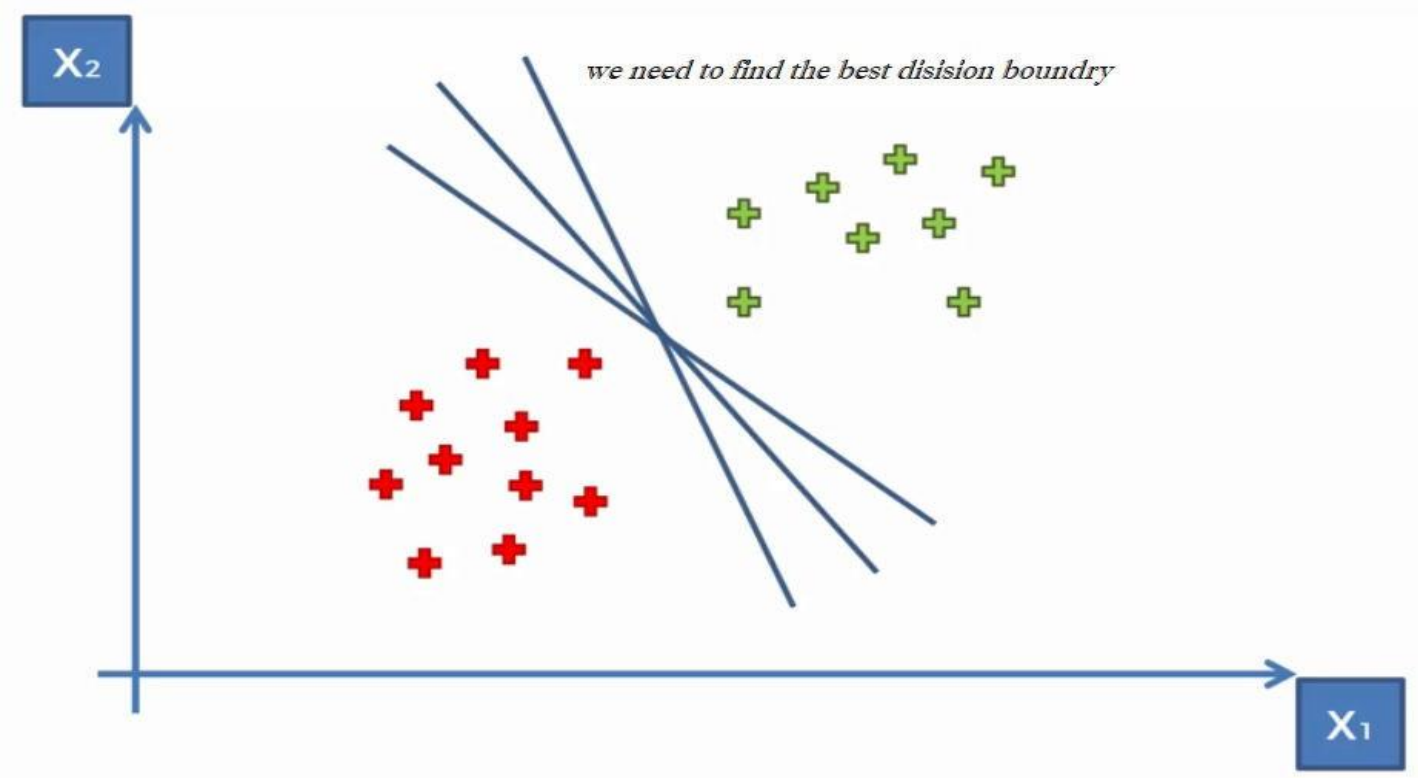
SVM Intuition

- Consider some usual points on a 2 dimensional space with two columns x1 & x2. How can we derive a line that will separate these two different points and classify them separately?



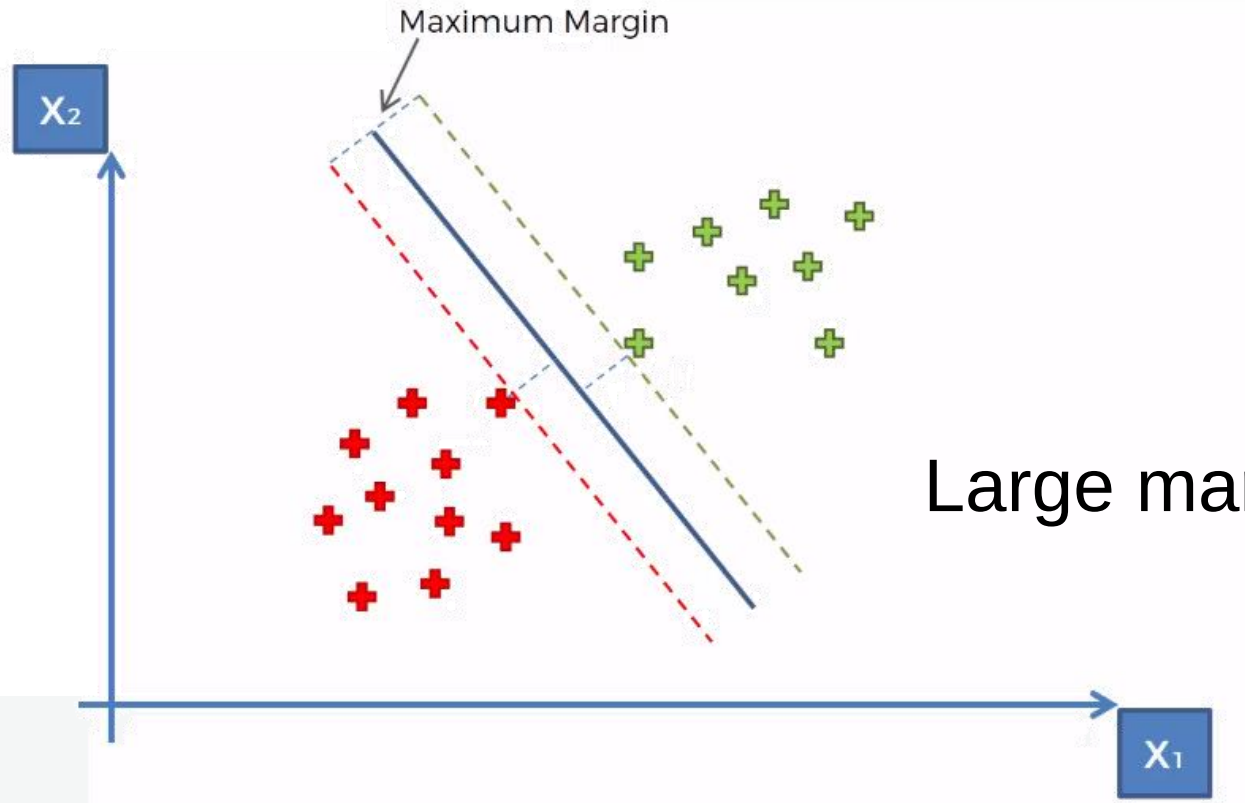
SVM Intuition

- Well there can be numerous ways of drawing lines in between that will achieve the same result as shown.



SVM Intuition

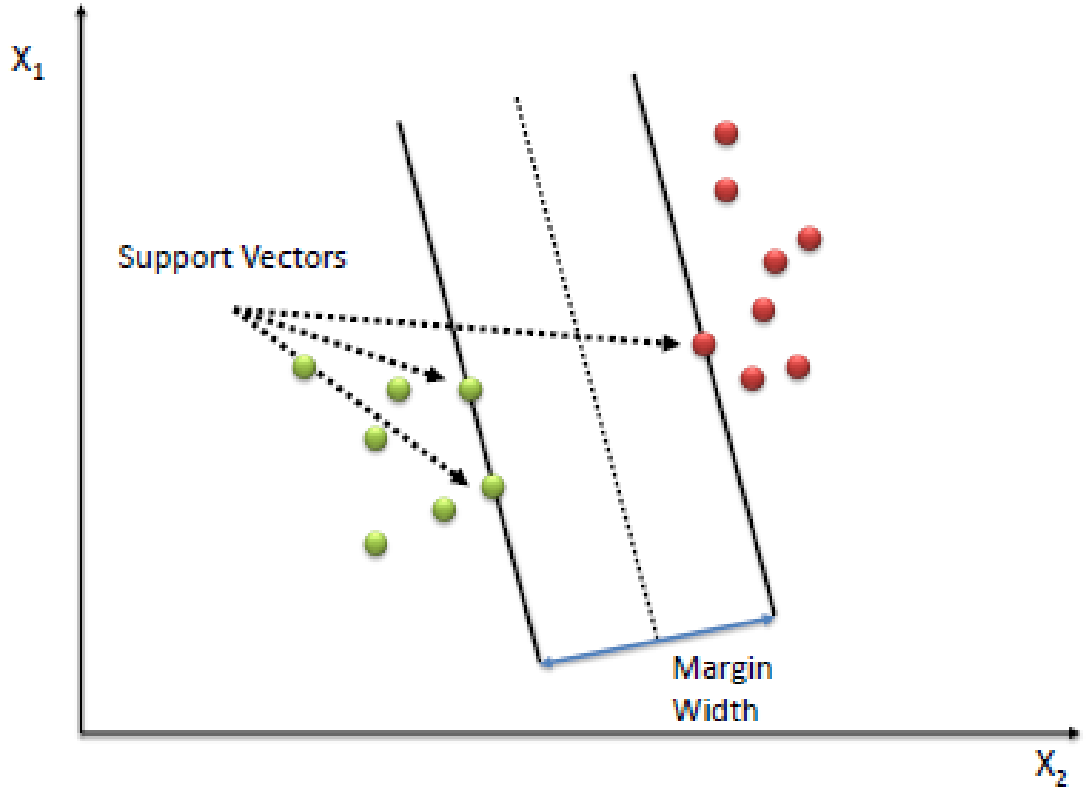
- SVM are about finding the **best decision boundary** that will help us to separate out space into classes. The required line is searched through **Maximum Margin.**



Large margin classifier

SVM Intuition

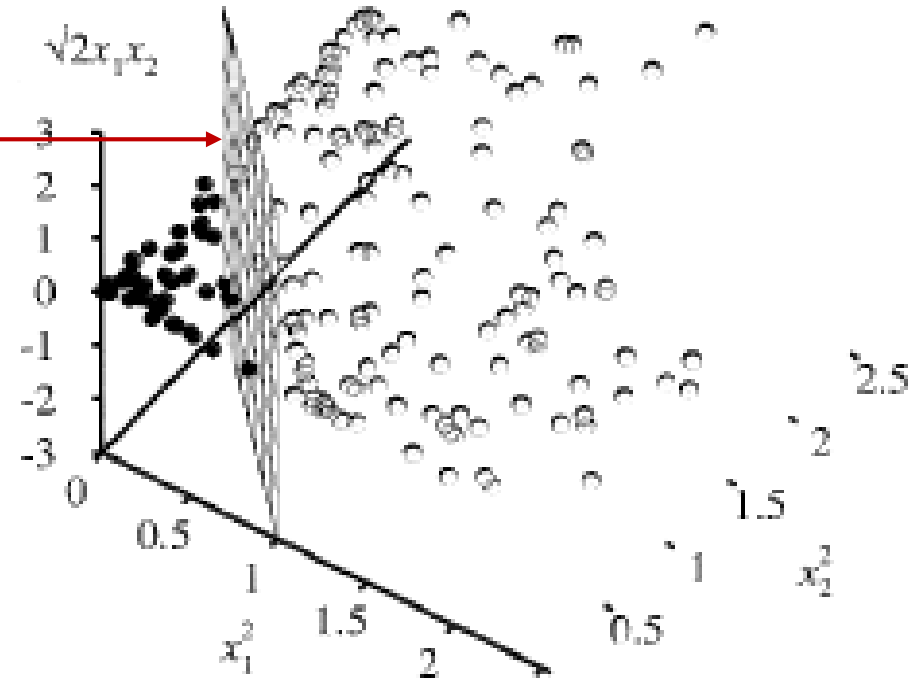
- ***Maximum Margin*** means that the distance between the line and each of these points (touching Red and Green point) is equidistant. The boundary points are know as **Support Vectors**.



SVM Intuition

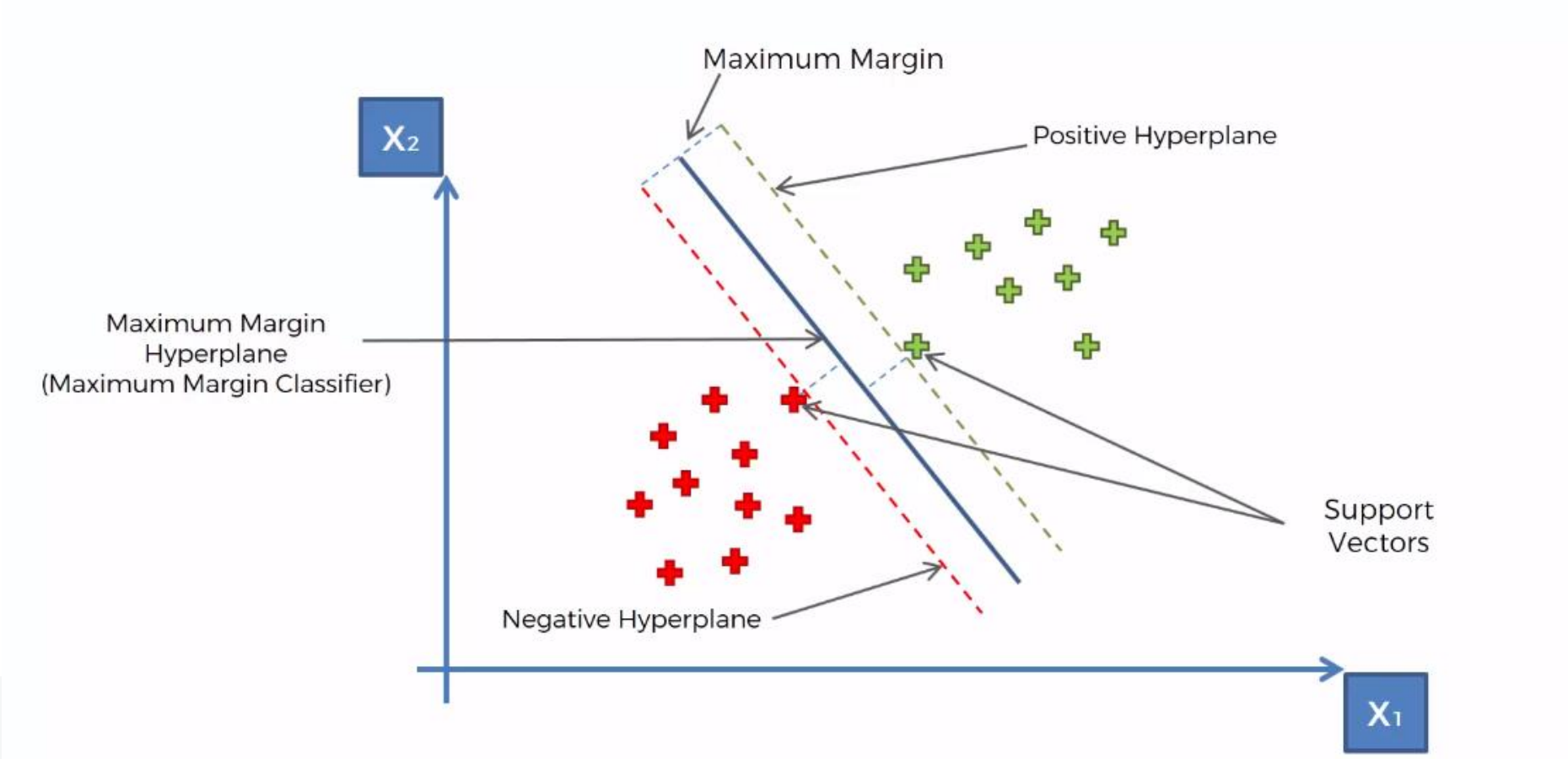
- Now we have got the line in middle which is called **the maximum margin Hyperplane** or the **maximum margin classifier**. In 2D space its just like a classifier like a line but in multidimensional space its a Hyperplane.

Maximum margin hyperplane
(maximum margin classifier)



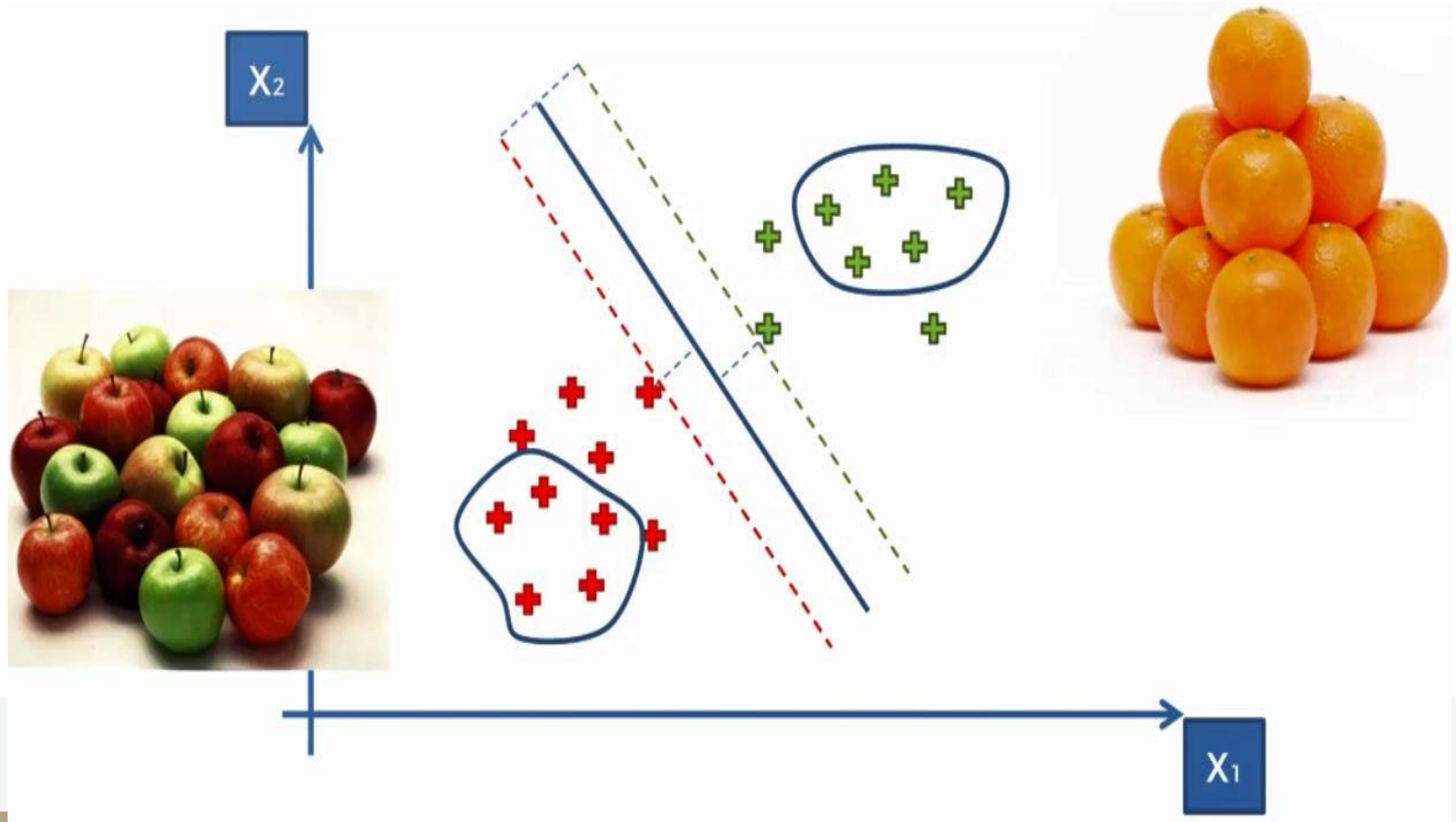
SVM Intuition

So the boundary touching to Green one is called the **Positive Hyperlane** and the boundary touching to Red is called the **Negative Hperplane**.



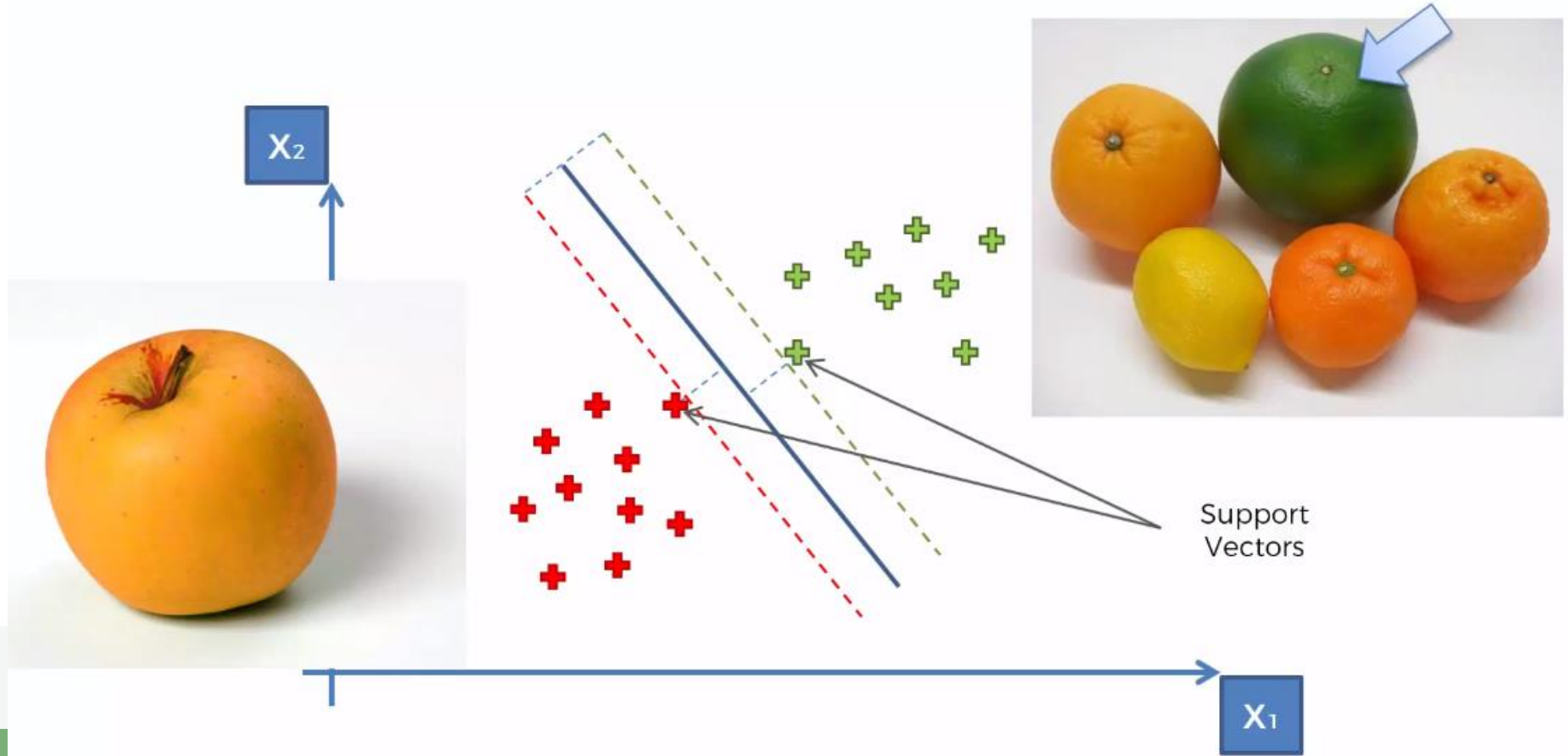
Why SVM is difference?

Predominantly a Machine Algorithm try to learn from apples that are very like apples and same for orange. So it would know what an apple and what an orange is and that's how most of the Machine Learning algorithms work and based on that it will be able to make some predictions.



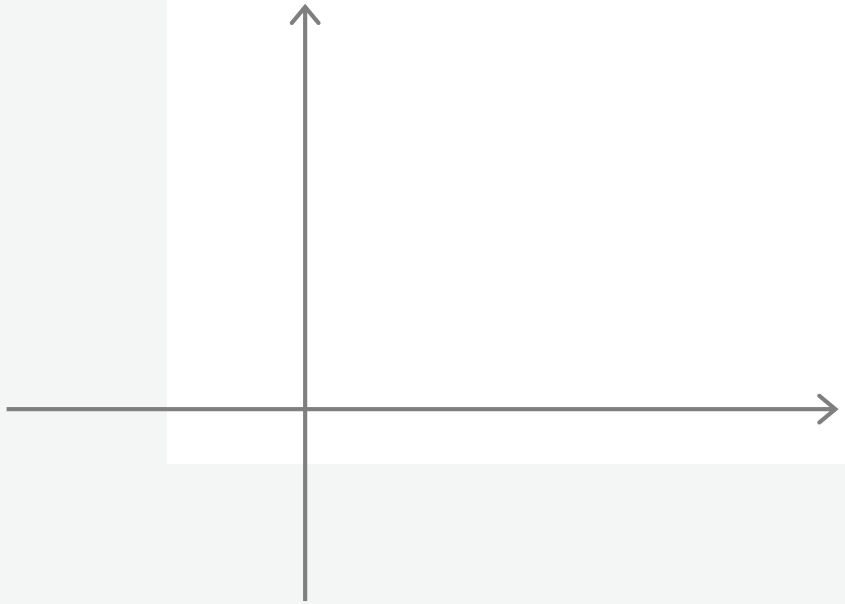
Why SVM is difference?

In SVM, it's slight different. E.g in apple category it tries to find out an apple which more looks like an orange and in orange category it tries to find out an orange which more looks like an apple. Those two things becomes **'Support Vectors**' which are close to the 'Maximum Margin'.



Vector Inner Product

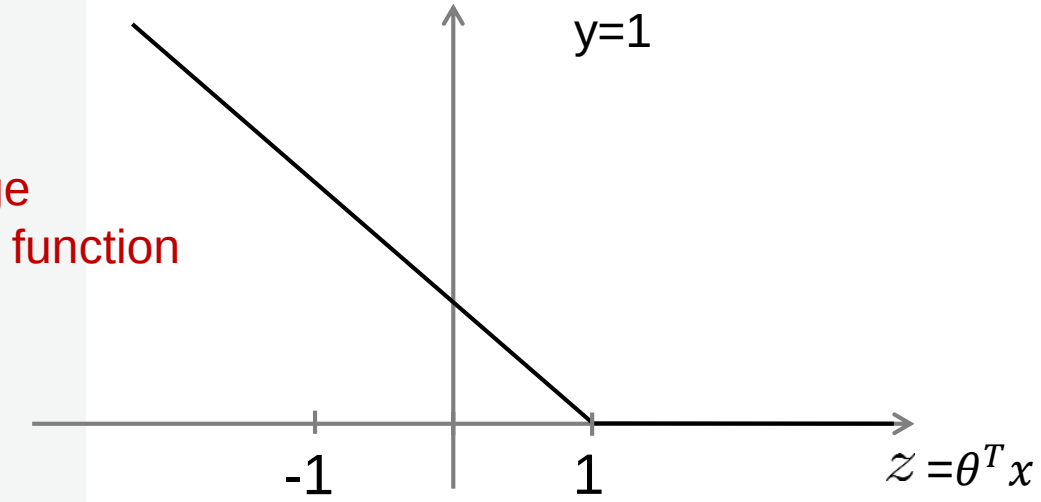
$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

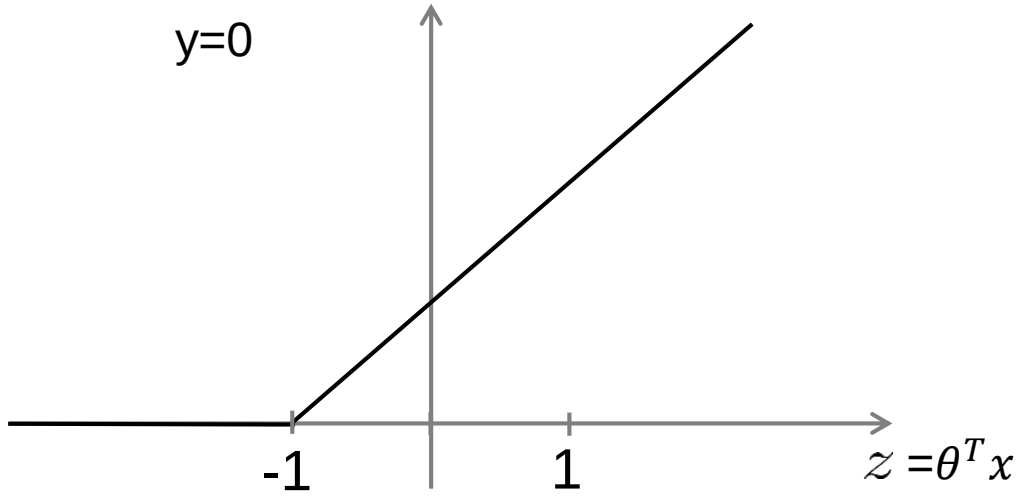


Support Vector Machine

$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

hinge
loss function





If $y = 1$, we want $\theta^T x \geq 1$ (not just ≥ 0)

If $y = 0$, we want $\theta^T x \leq -1$ (not just < 0)

That means, every datapoint x must have a distance to the hyperplane larger than the margin

Support Vector Machine

$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

regularization parameter $(C) = \frac{1}{\lambda}$

Term (A)

Term (B)

If C is too large, e.g. **C**=100,000, and minimize this cost function, the term (**A**) will nearly 0. So, the cost function is equal term **B**

$$\min_{\theta} = \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

Whenever $\theta^T x \geq 1$ $y^{(i)} = 1$

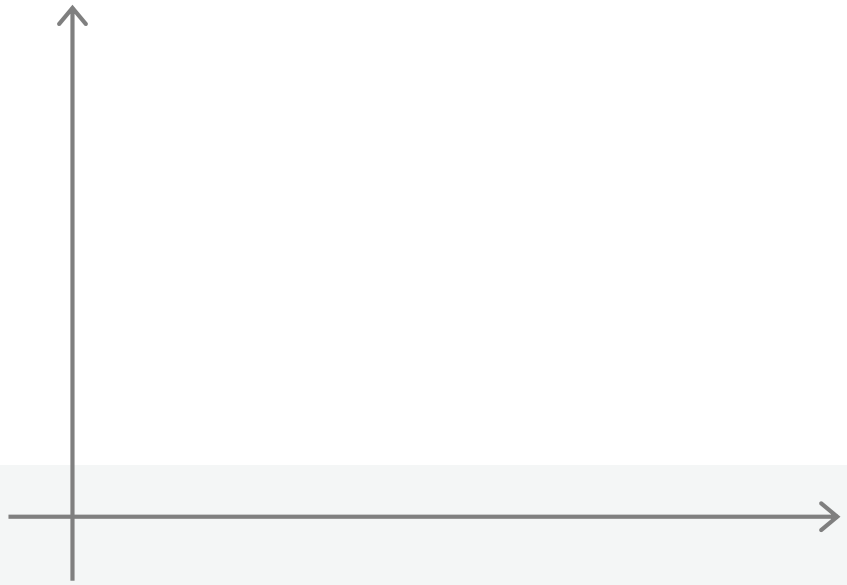
$\theta^T x \leq -1$ $y^{(i)} = 0$

SVM Decision Boundary

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

$$\text{s.t.} \quad \theta^T x^{(i)} \geq 1 \qquad \text{if } y^{(i)} = 1$$

$$\theta^T x^{(i)} \leq -1 \qquad \text{if } y^{(i)} = 0$$



SVM Decision Boundary

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

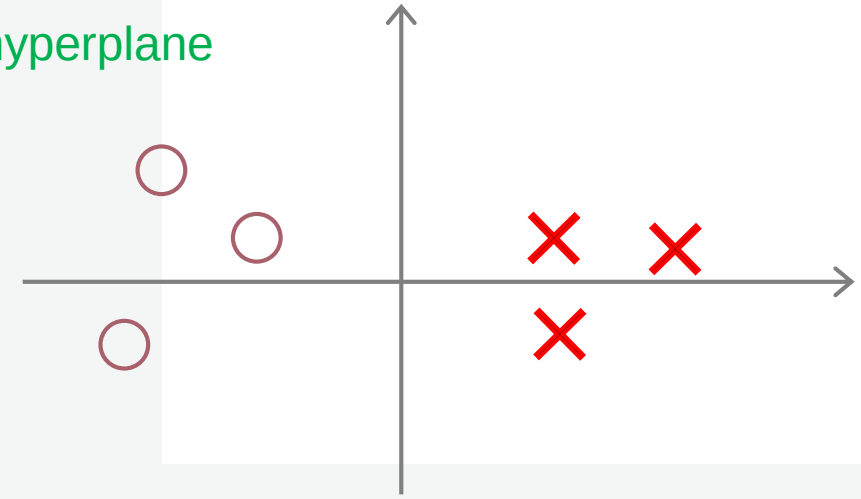
$$\text{s.t.} \quad p^{(i)} \cdot \|\theta\| \geq 1 \qquad \text{if } y^{(i)} = 1$$

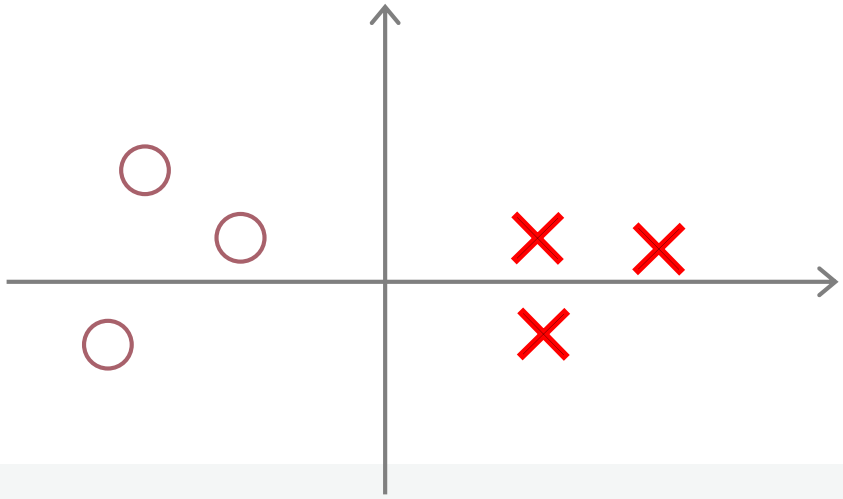
$$p^{(i)} \cdot \|\theta\| \leq -1 \qquad \text{if } y^{(i)} = 1$$

where $p^{(i)}$ is the projection of $x^{(i)}$ onto the vector θ.

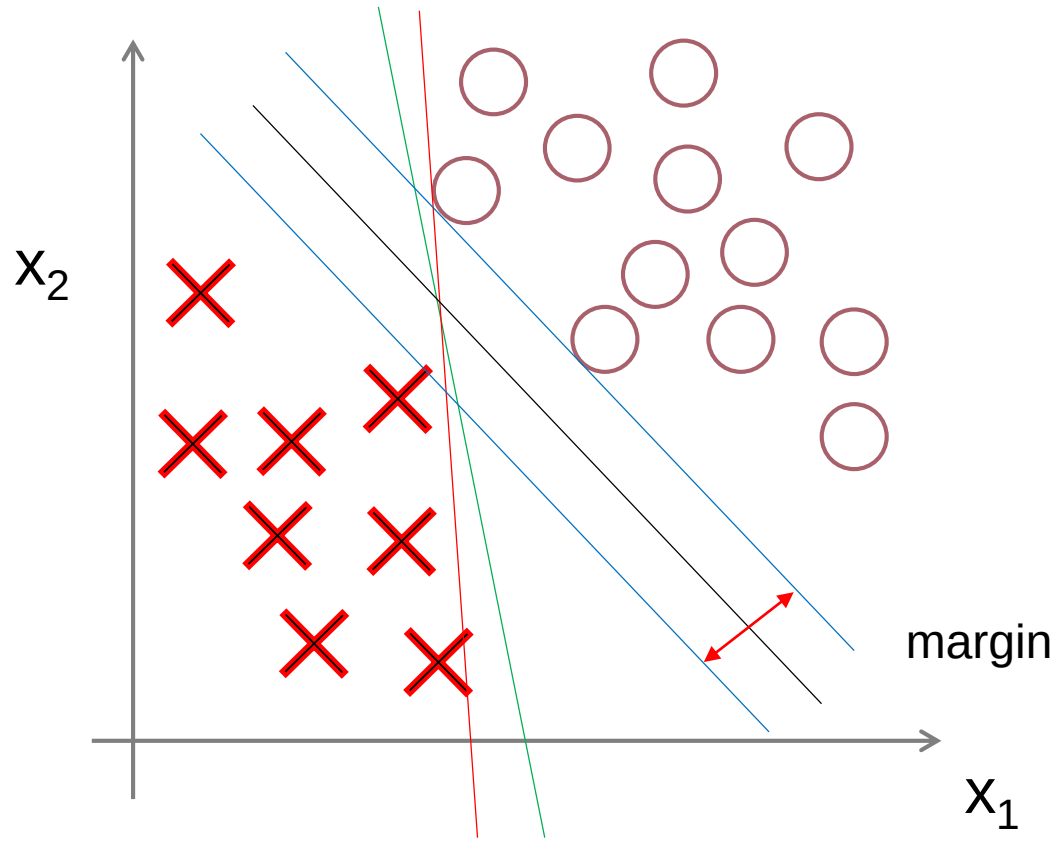
Simplification: $\theta_0 = 0$

hyperplane



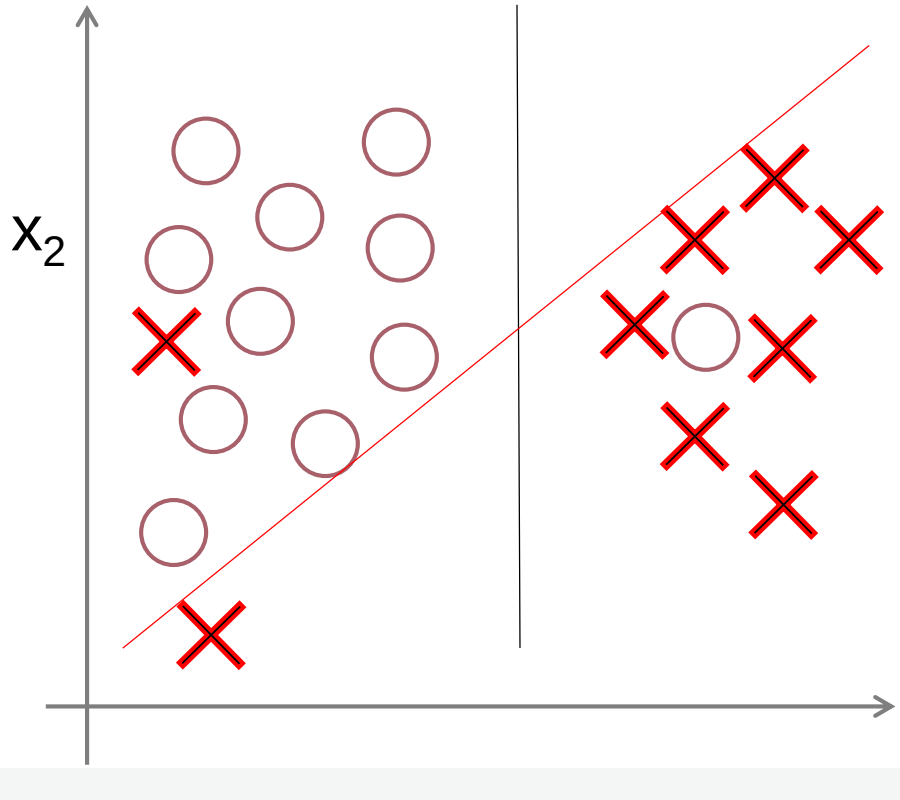


SVM Decision Boundary: Linearly separable case



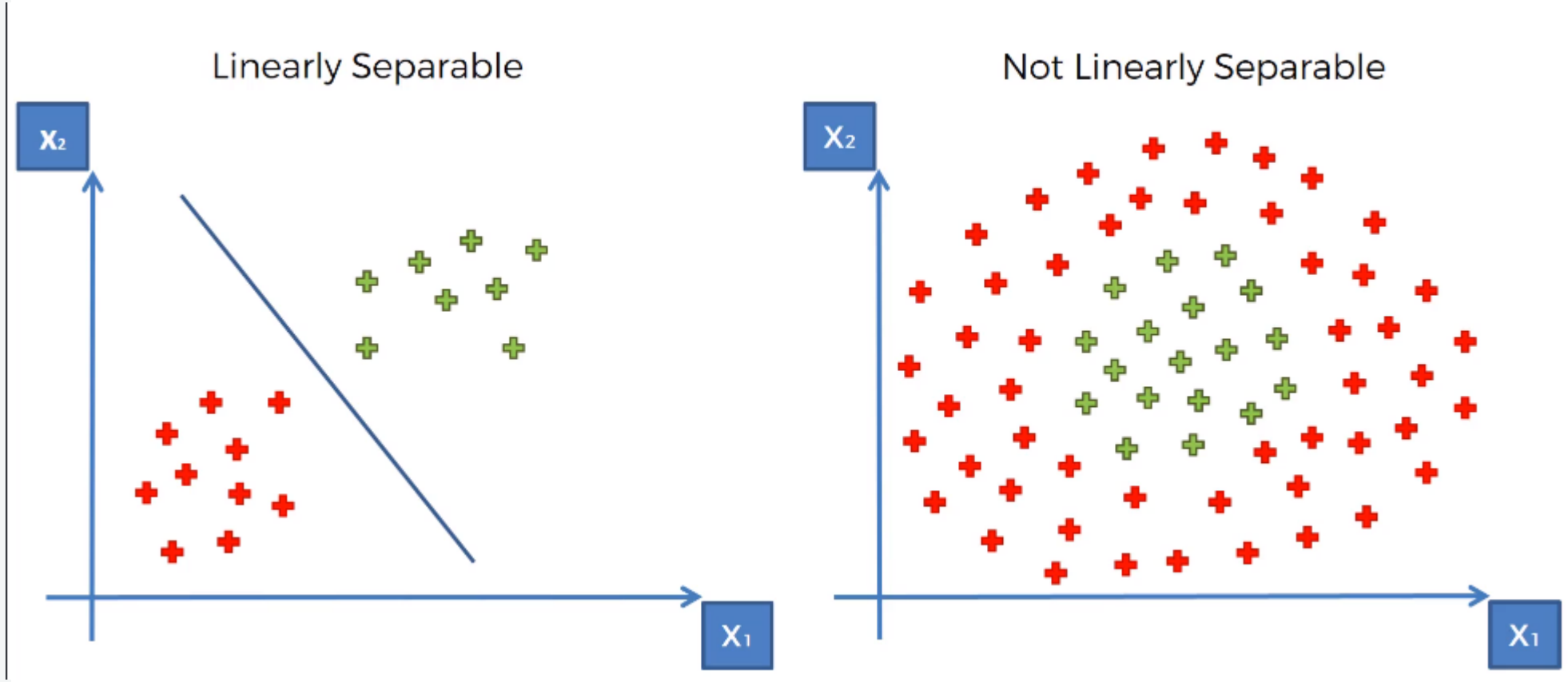
Large margin classifier in presence of outliers

In presence of outlier and C is large, What is SVM decision?
And if C is small, What is SVM decision?



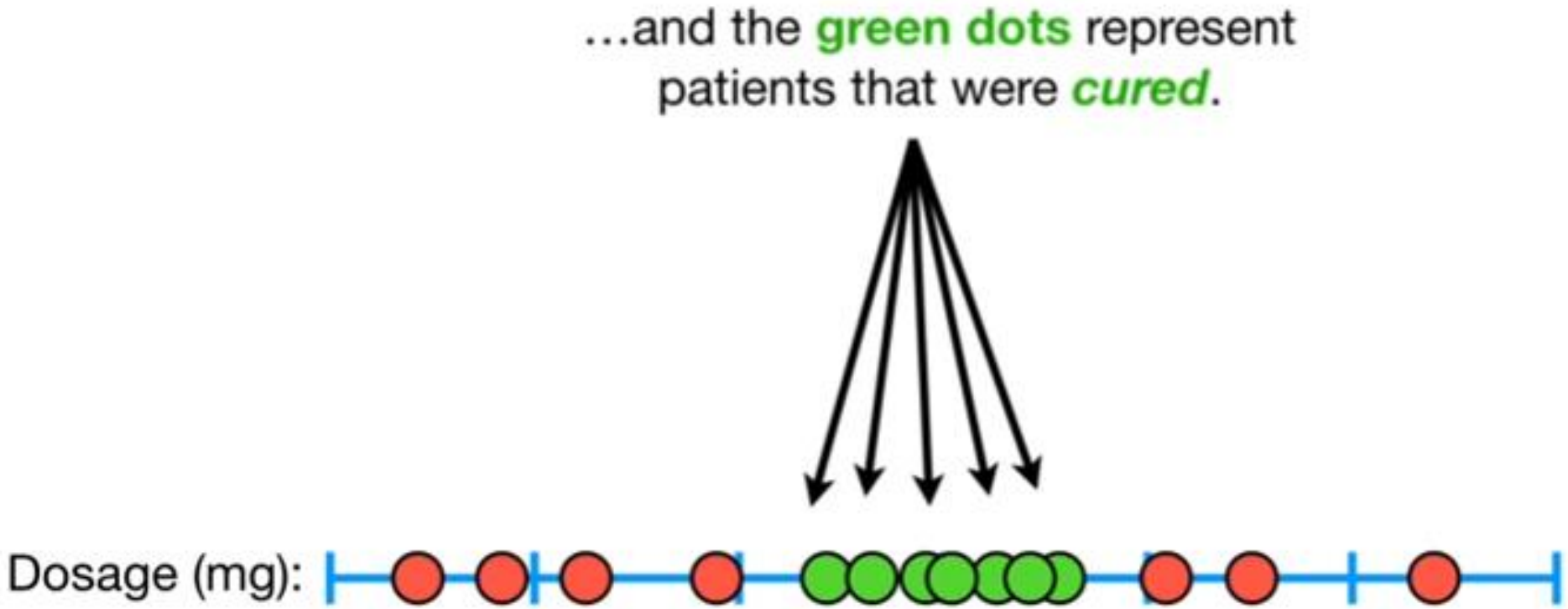
When we allow misclassifications, the distance between the observations and the threshold is called a **Soft Margin**.

Linear separability

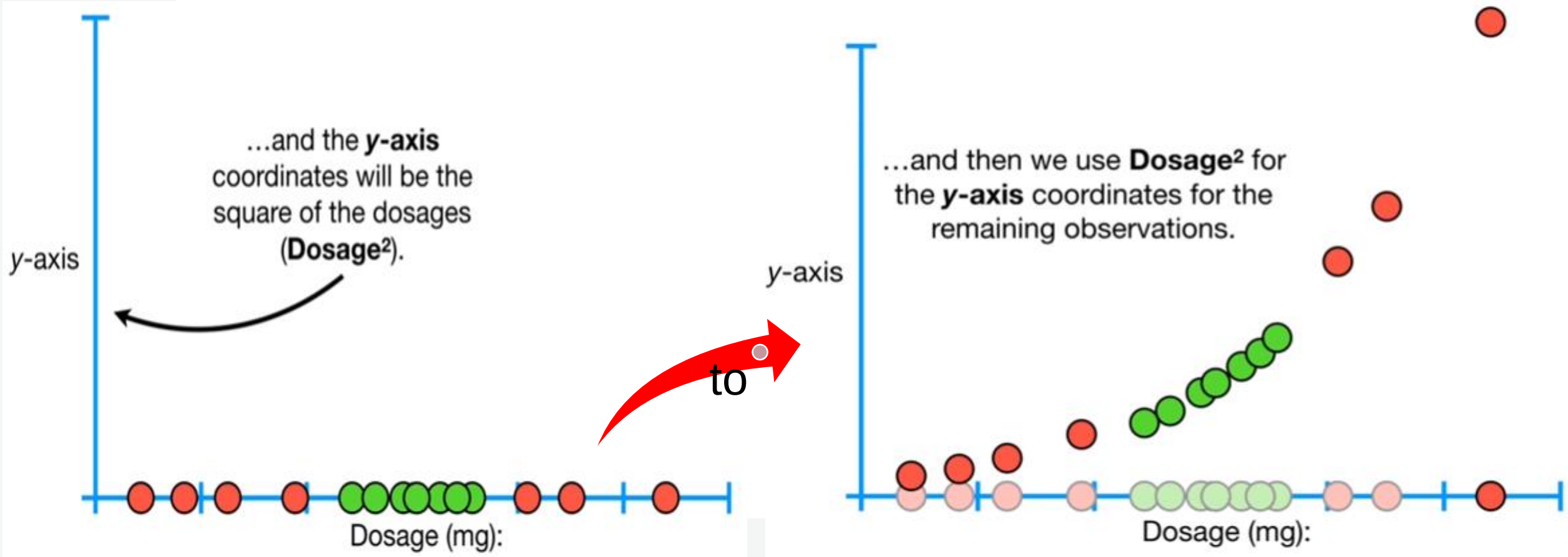


In 1D??

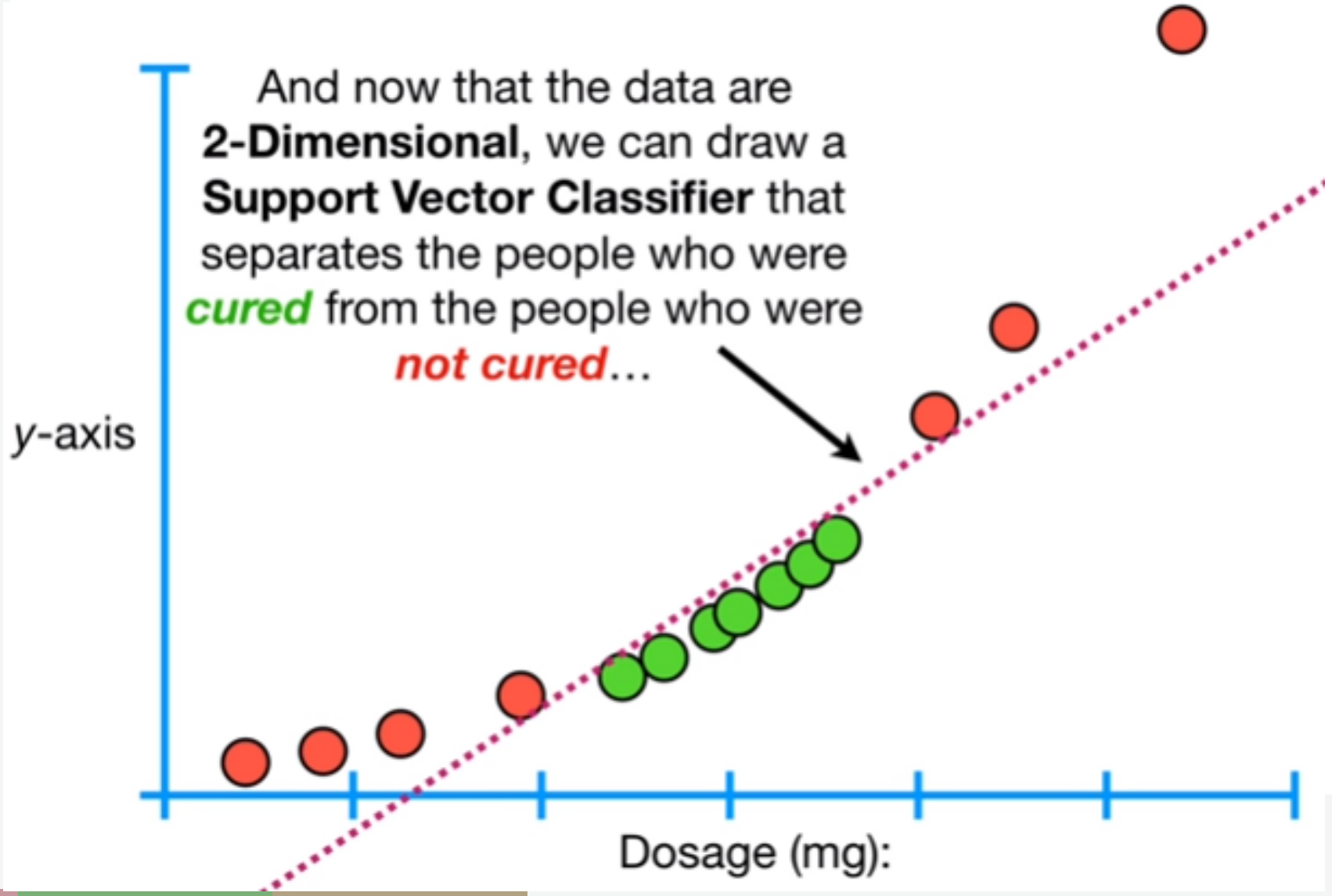
- Consider the drug dosage and patient classification as cured (green dots) or not cured (red dots). In other words, the drug doesn't work if the dosage is too small or too large



Solution: Mapping a higher dimension

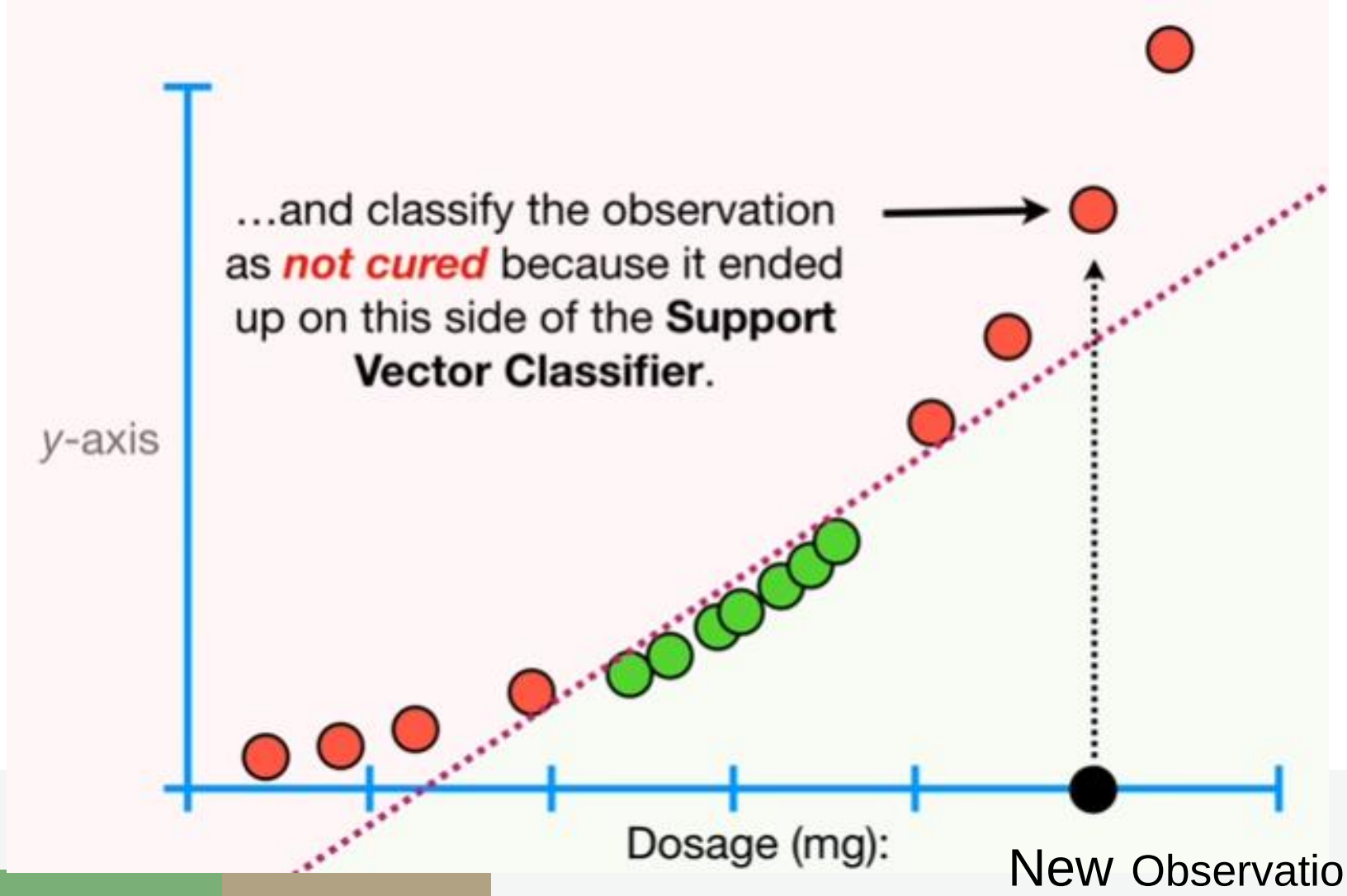


Linearly Separable on 2-Dimension



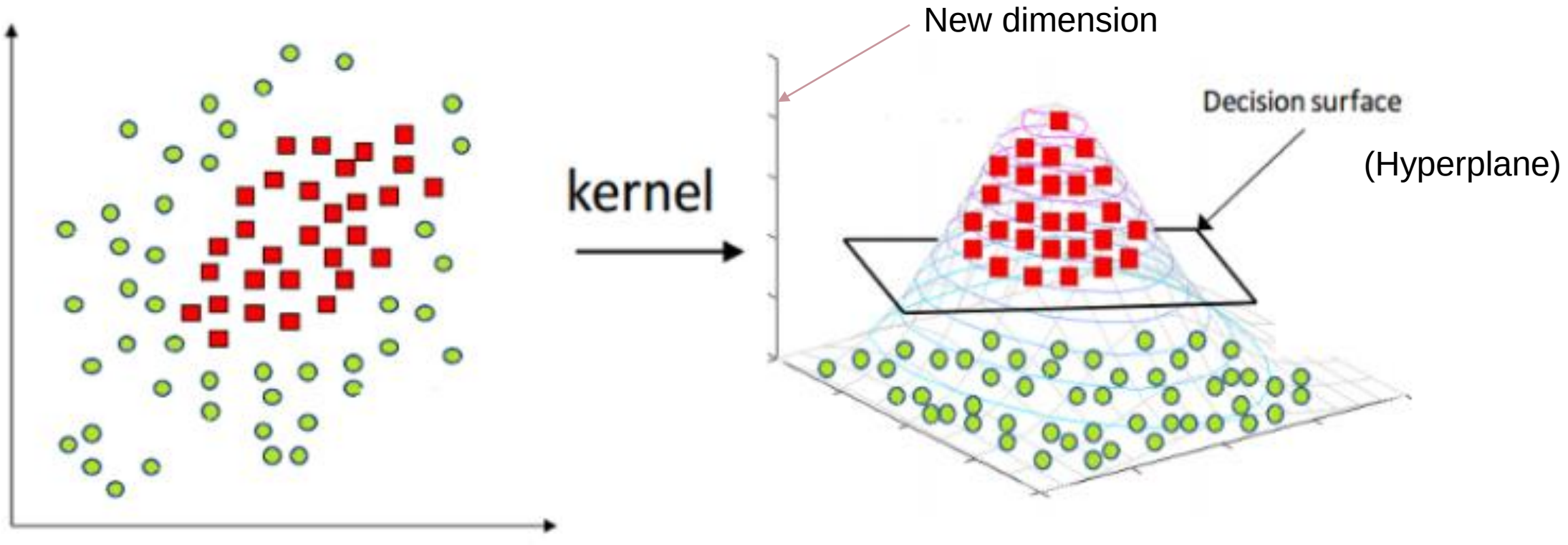
Linearly Separable on 2-Dimension

Polynomial kernel with 2 degree



If data was initially at 2D Space?

Solution = Mapping a higher dimension



Polynomial Kernel

The polynomial kernel systematically increases dimensions by setting **D** (i.e. the degree of the polynomial) and the relationships between each pair of observations are used to find a support vector classifier.

Note: We can find a good value for D with **cross validation.**

Kernel Functions

In order to make the mathematics possible, support vector machines use something called **kernel functions** to systematically find support vector classifier in higher dimensions.

Types of kernel functions:

1) Linear kernel
2) Polynomial kernel
3) Radial basis function kernel (RBF)/ Gaussian Kernel

Kernel Trick

Kernel functions only calculate the relationships between every pair of points as if they are in the higher dimensions they don't actually do the transformation this trick calculating the

high-dimensional relationships without actually transforming the data to the higher dimension is called the **kernel trick**.

The kernel trick reduces the amount of computation required for

support vector machines by avoiding the math that transforms the data from low to high dimensions and it makes calculating the relationships in the infinite dimensions used by the radial kernel possible.

Polynomial Kernel Equation

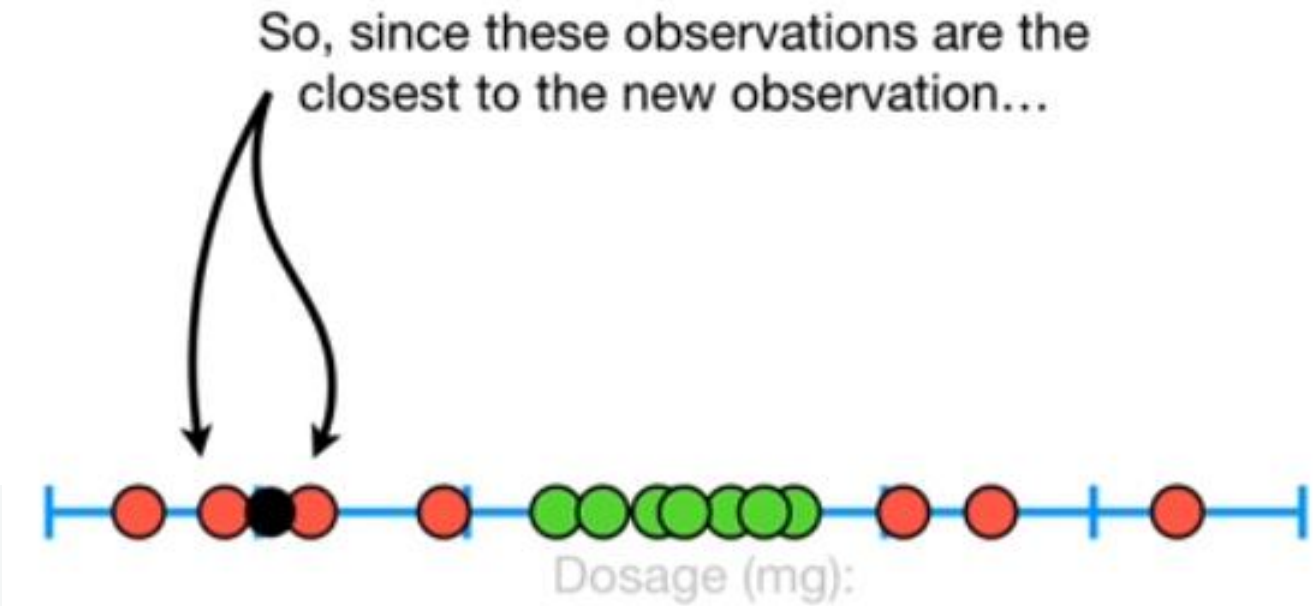
If we use the **kernel function**, which is denoted as ***k(x, y),*** instead of doing the complicated computations to transform the inputs from 3-dimensional space into 9-dimensional space for example, we reach the same result the by calculating the dot product of x -transpose and y.

Polynomial kernel is: $K(x, y) = (x^\top y + c)^d$

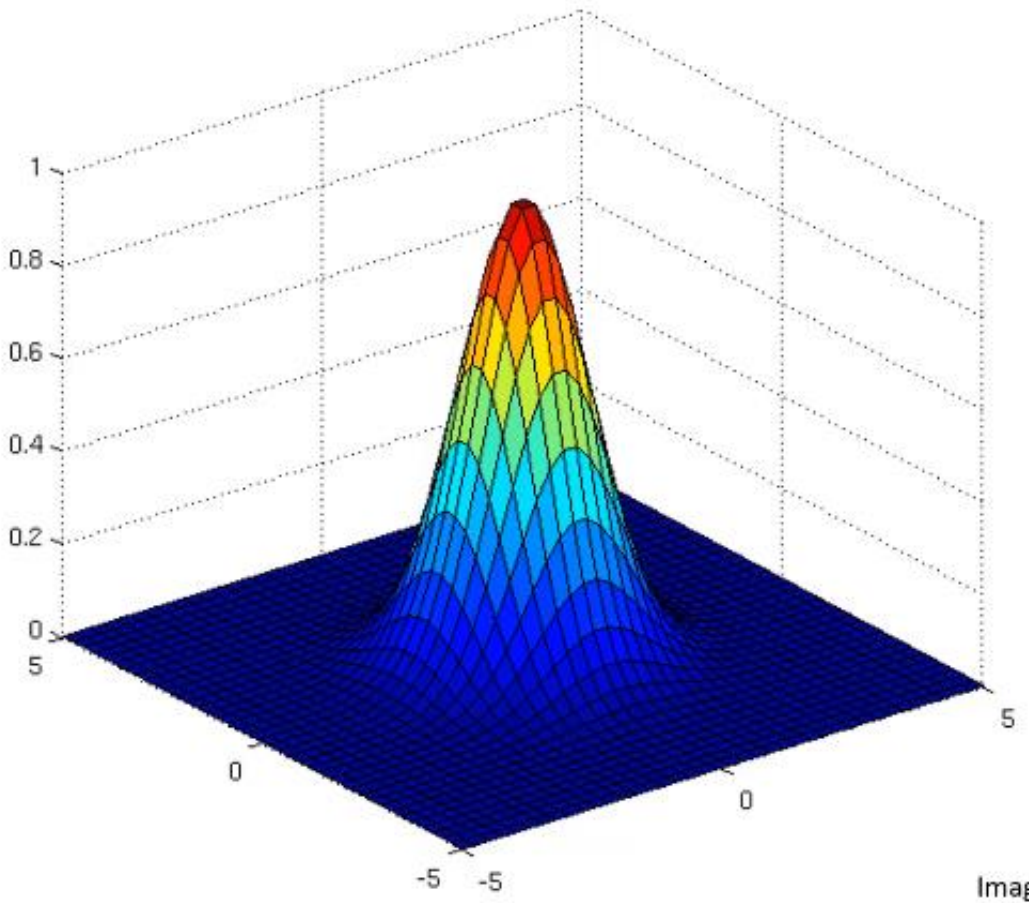
where *x* and *y* are vectors in the *input space*, i.e. vectors of features computed from training or test samples, d is the polynomial degree and $c \geq 0$ is a free parameter trading off the influence of higher-order versus lower-order terms in the polynomial.

Radial basis function kernel (RBF)

- Radial basis function kernel unfortunately finds support vector classifiers in infinite dimensions. But, when using a new observation, the radial kernel behaves like a **weighted nearest neighbor model.** In other words, the closest observations have a lot of influence on how we classify the new observation and observations that are further away have relatively little influence on the classification.



RBF kernel/Gaussian kernel



Given x , compute new feature depending on proximity to landmarks l^i

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

Where,

- $\vec{x}$: X vector (some point in our dataset)
- $\overrightarrow{l^i}$: l stands for landmarks (i represents the number of landmarks)
- $(\left\|\vec{x} - \overrightarrow{l^i}\right\|)^2$: Distance between a point and the landmark squared

Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

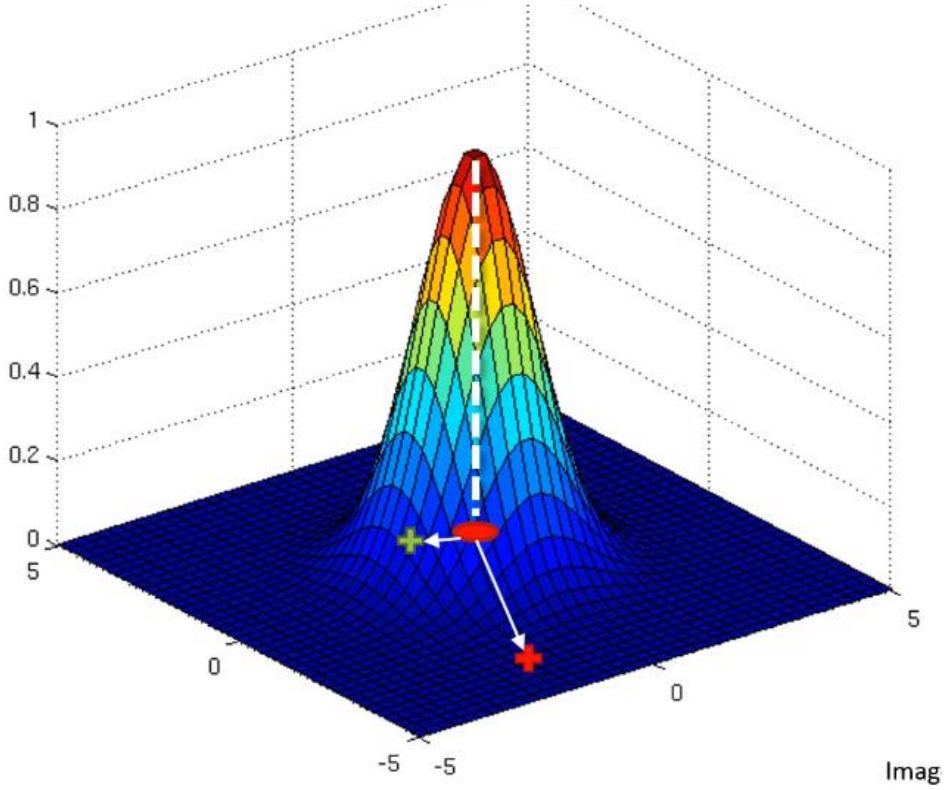
Kernels and Similarity

The landmark l^1 is placed directly at the center of the dataset and is used to calculate the distance to a point. In the picture below the landmark is represented by a red dot. The red plus sign (x1) and green plus sign (x2) represent different data points.

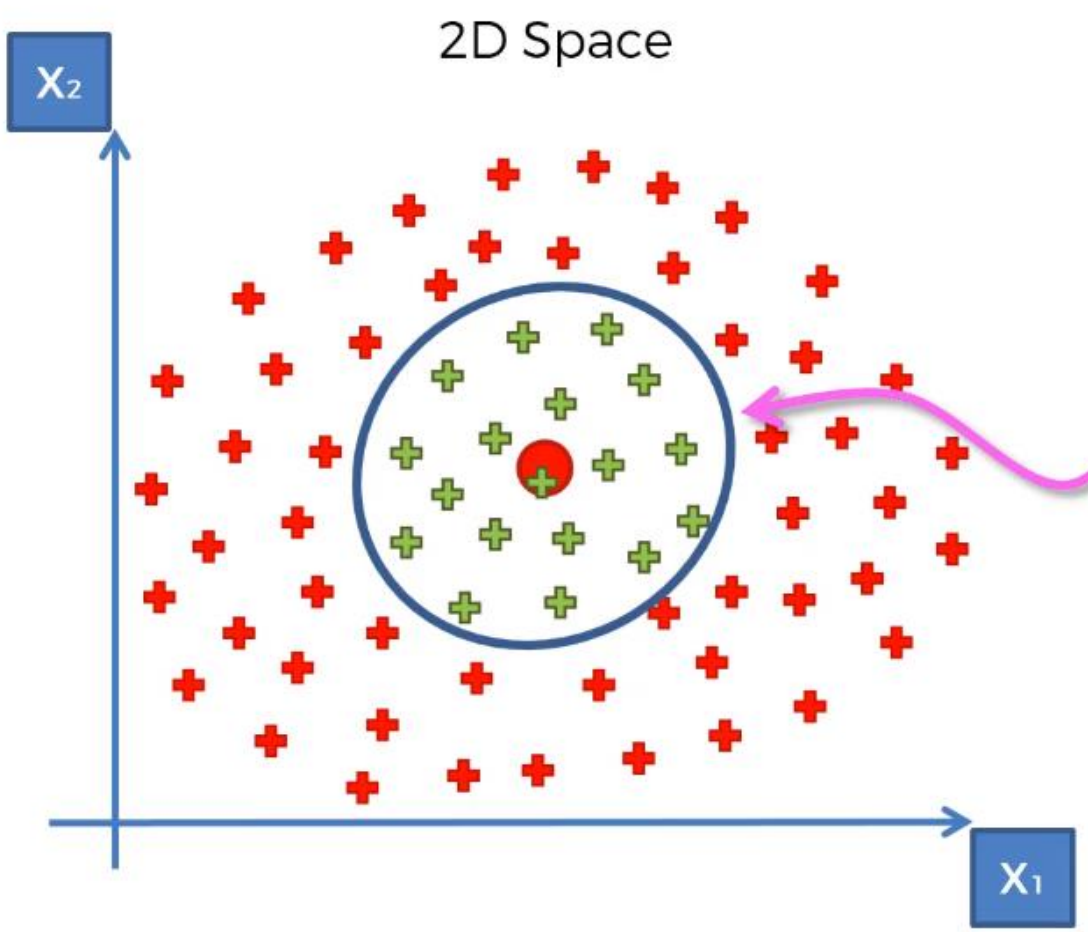
$$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

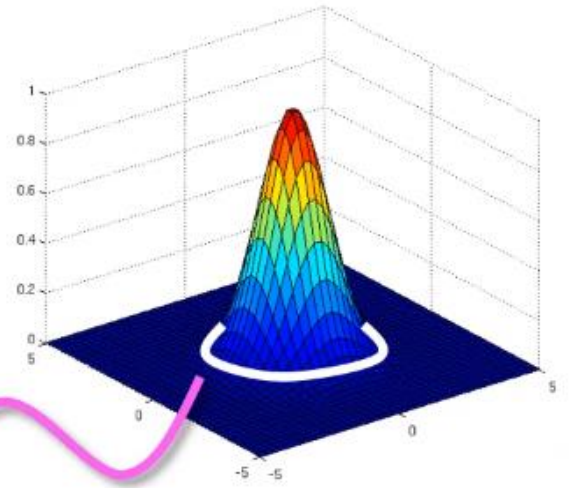
If $x_1 \approx l^{(1)}$:

If x_2 if far from $l^{(1)}$:



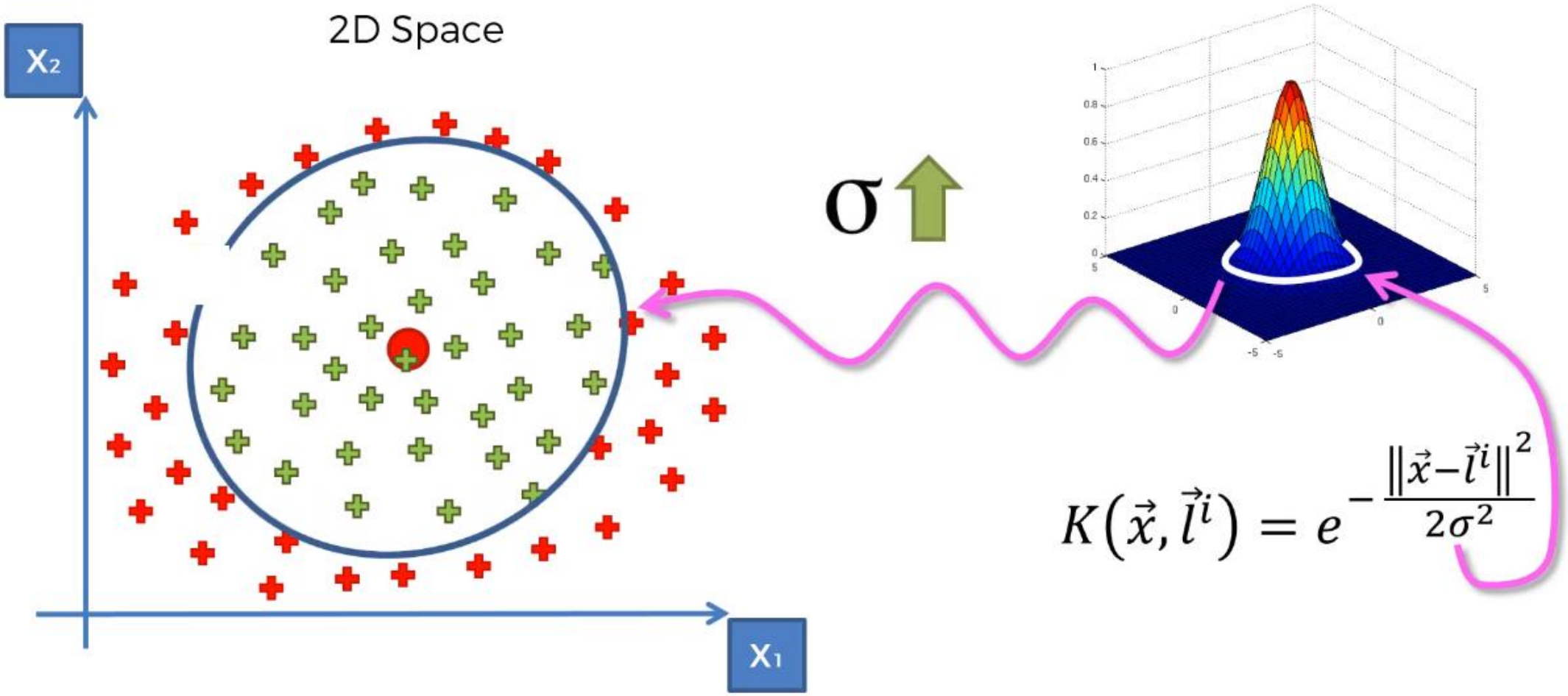
RBF kernel/Gaussian kernel



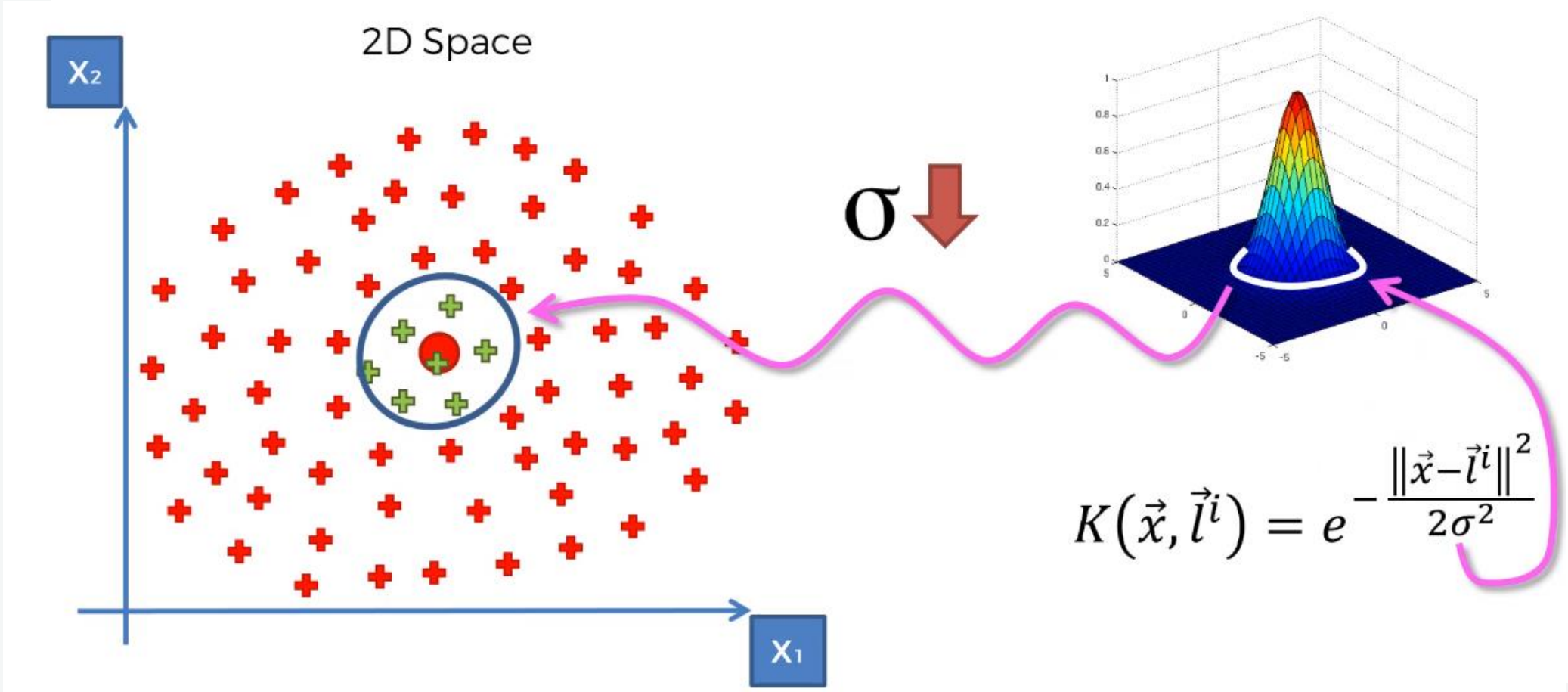


$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

RBF kernel/Gaussian kernel



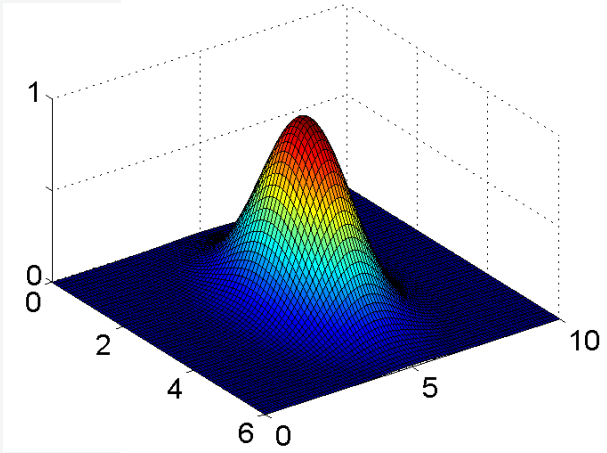
Gaussian Radial basis function kernel, RBF kernel

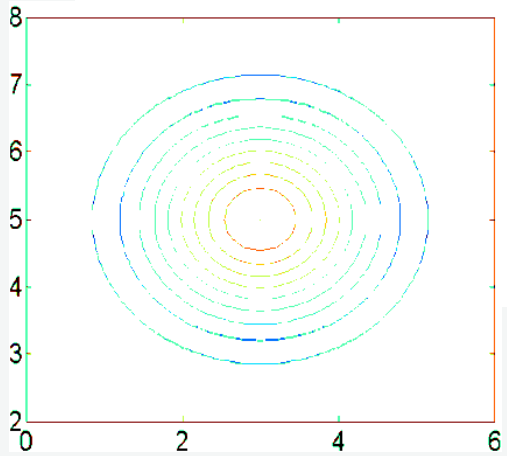


Example:

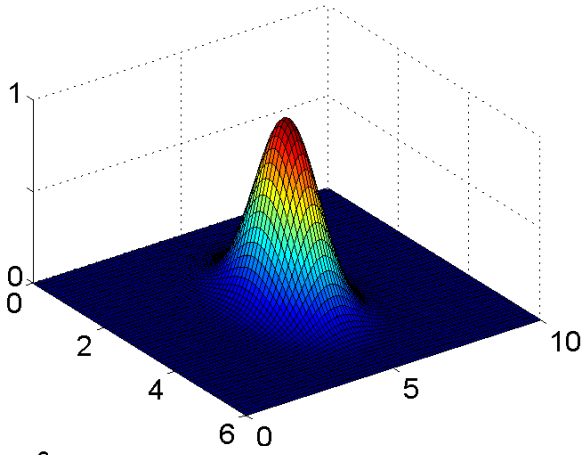
$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

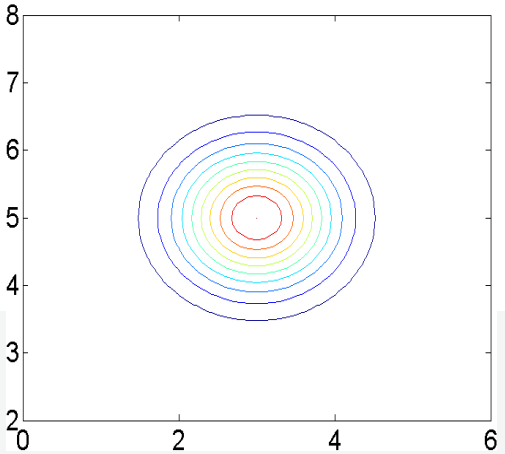
$\sigma^2 = 1$



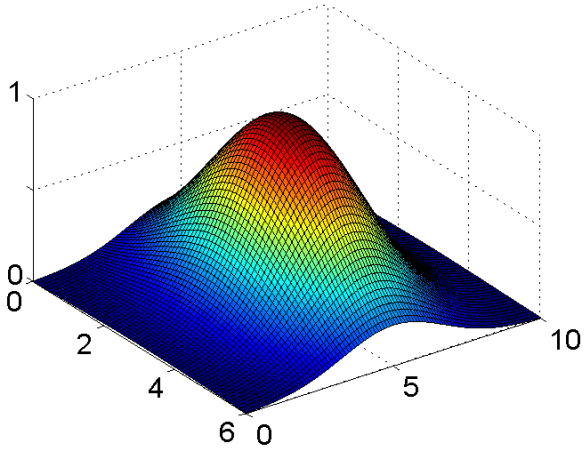


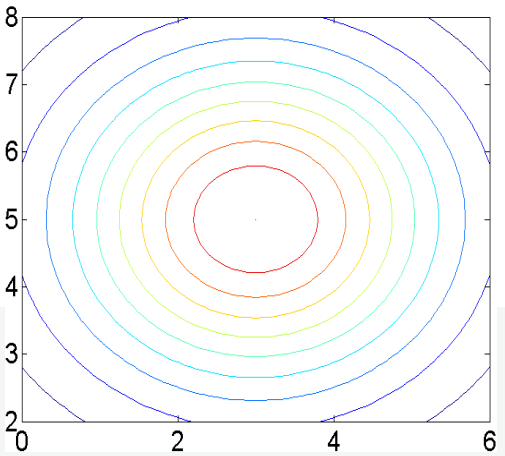
$\sigma^2 = 0.5$





$\sigma^2 = 3$





Thanks