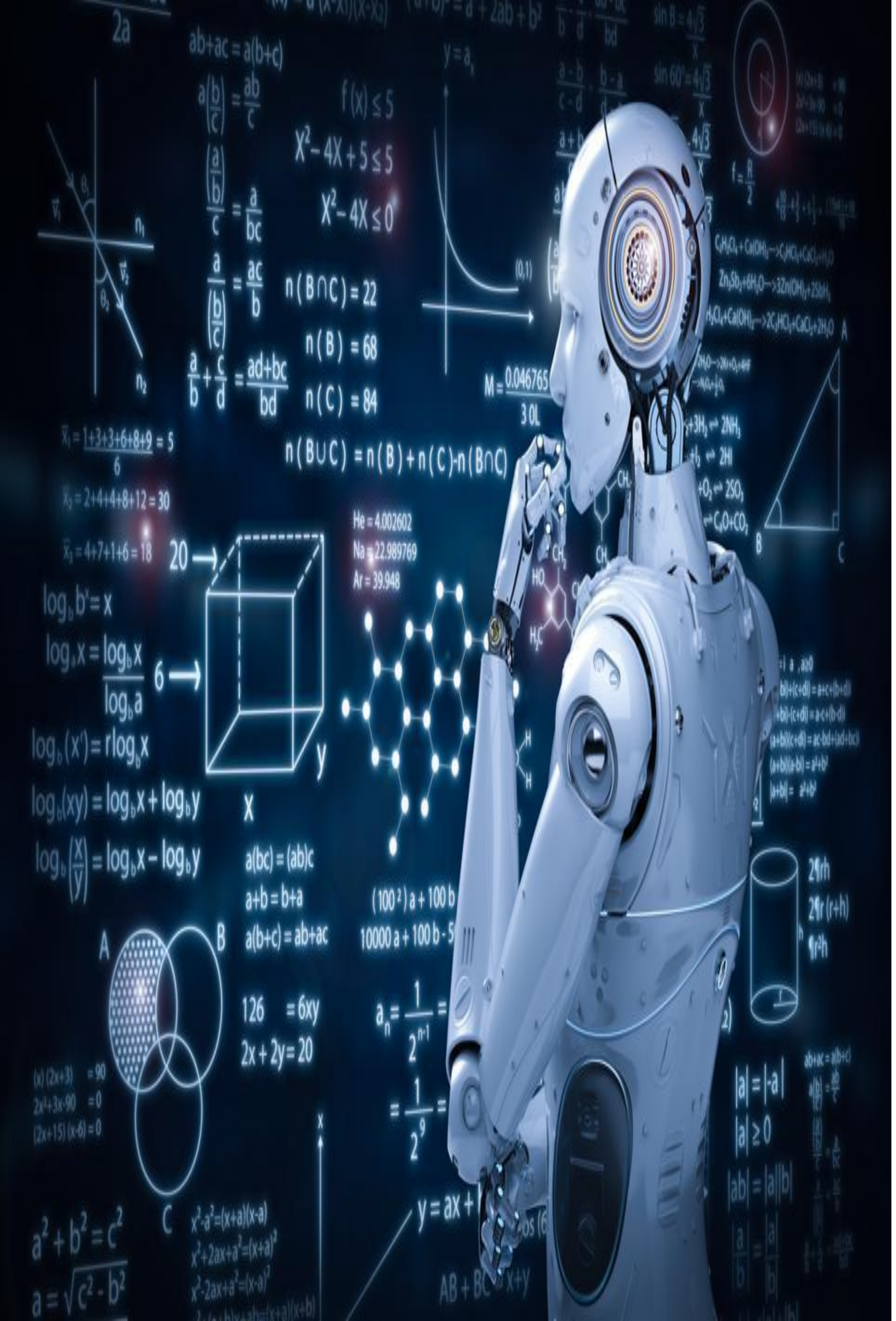


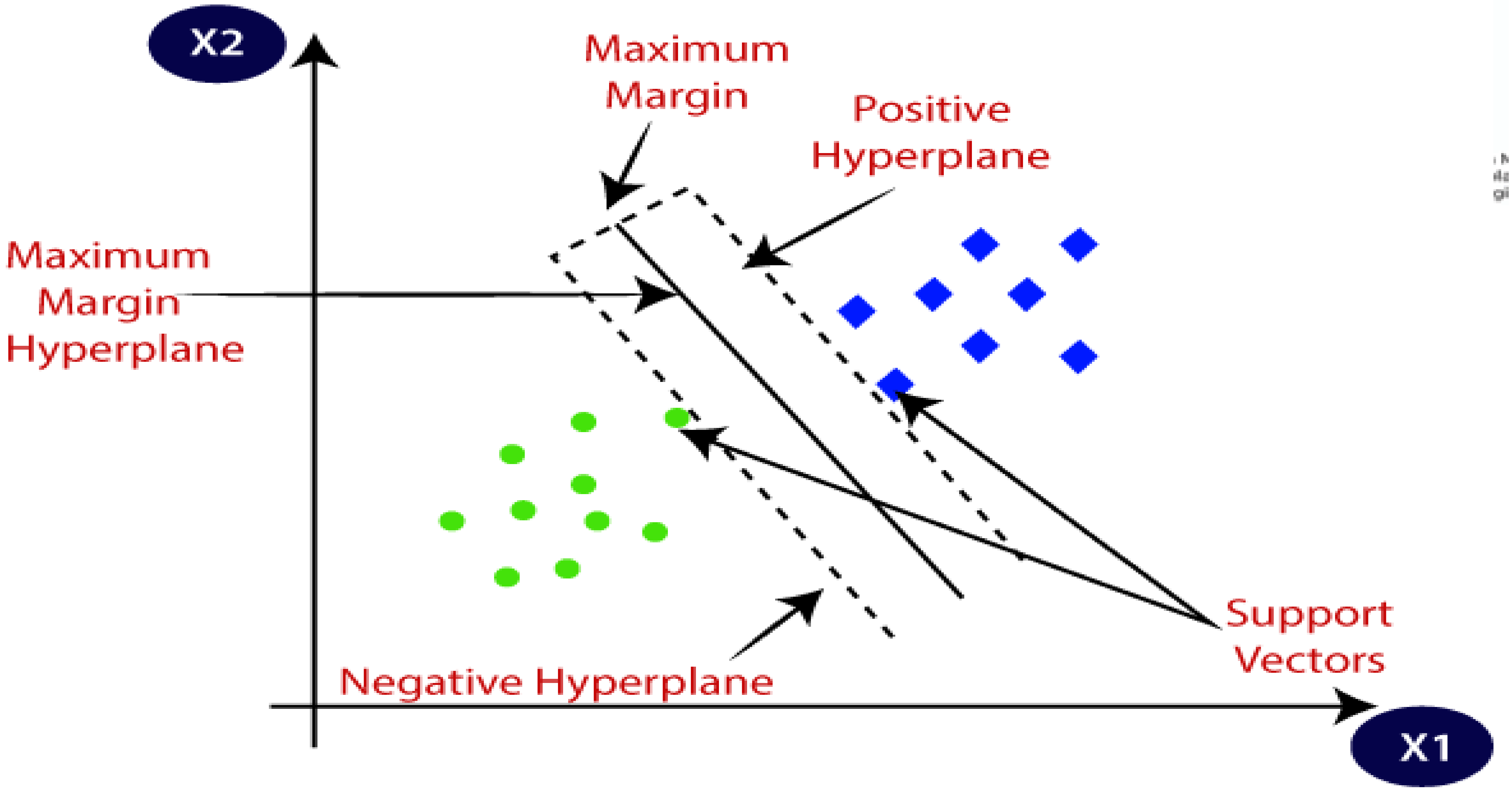
Practical Machine Learning

Day 13: Mar22 DBDA

Kiran Waghmare

Agenda

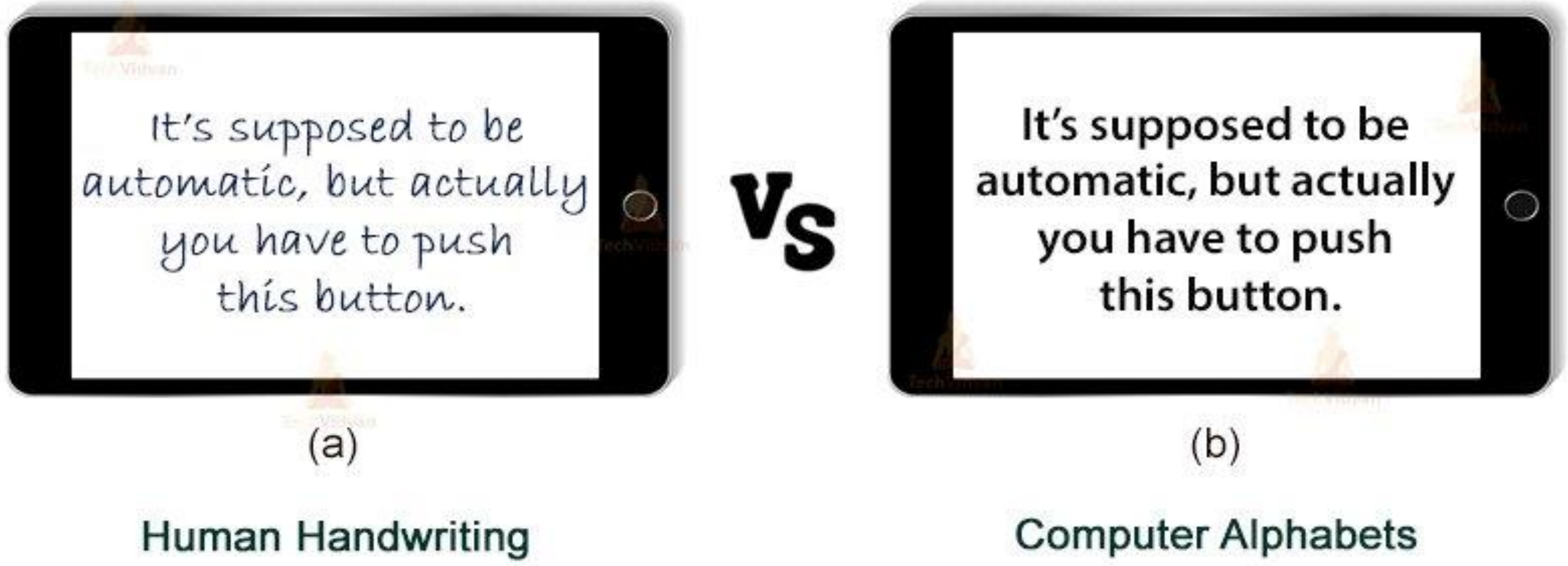
- SVM
- SVM-Kernel

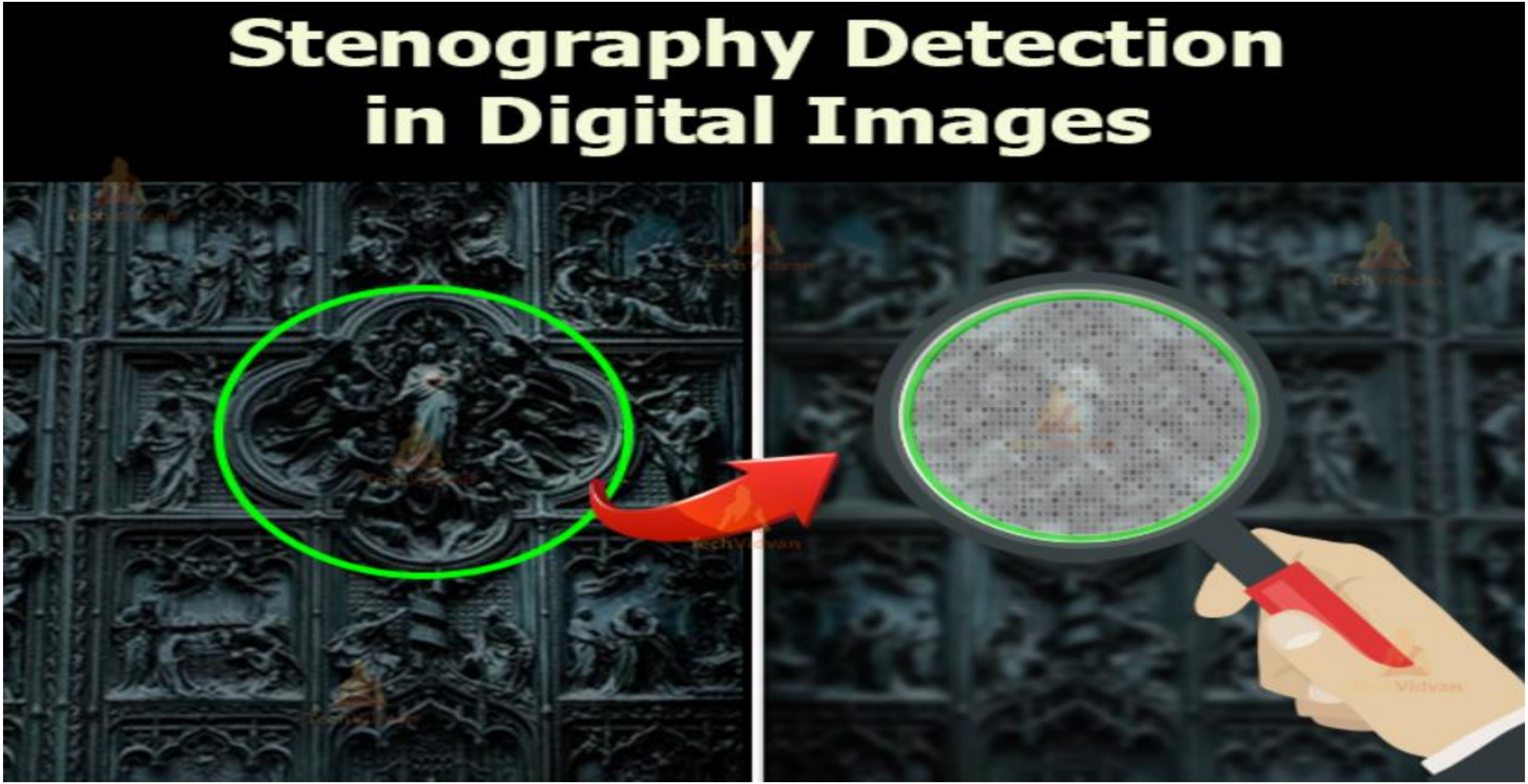

X2
Maximum
Margin
Positive
Hyperplane
Maximum
Margin
Hyperplane
Negative Hyperplane
Support
Vectors
X1

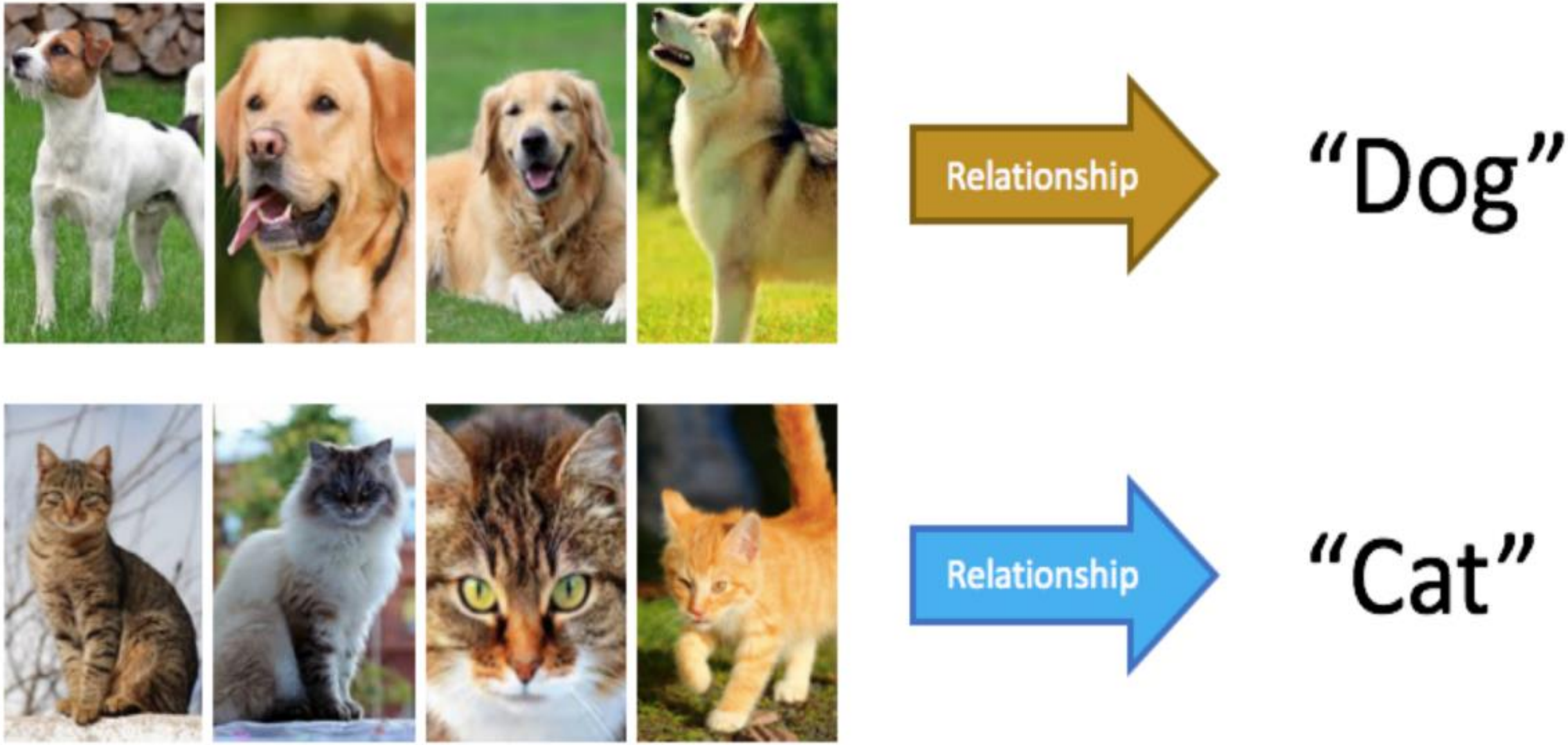
Support Vector Machine

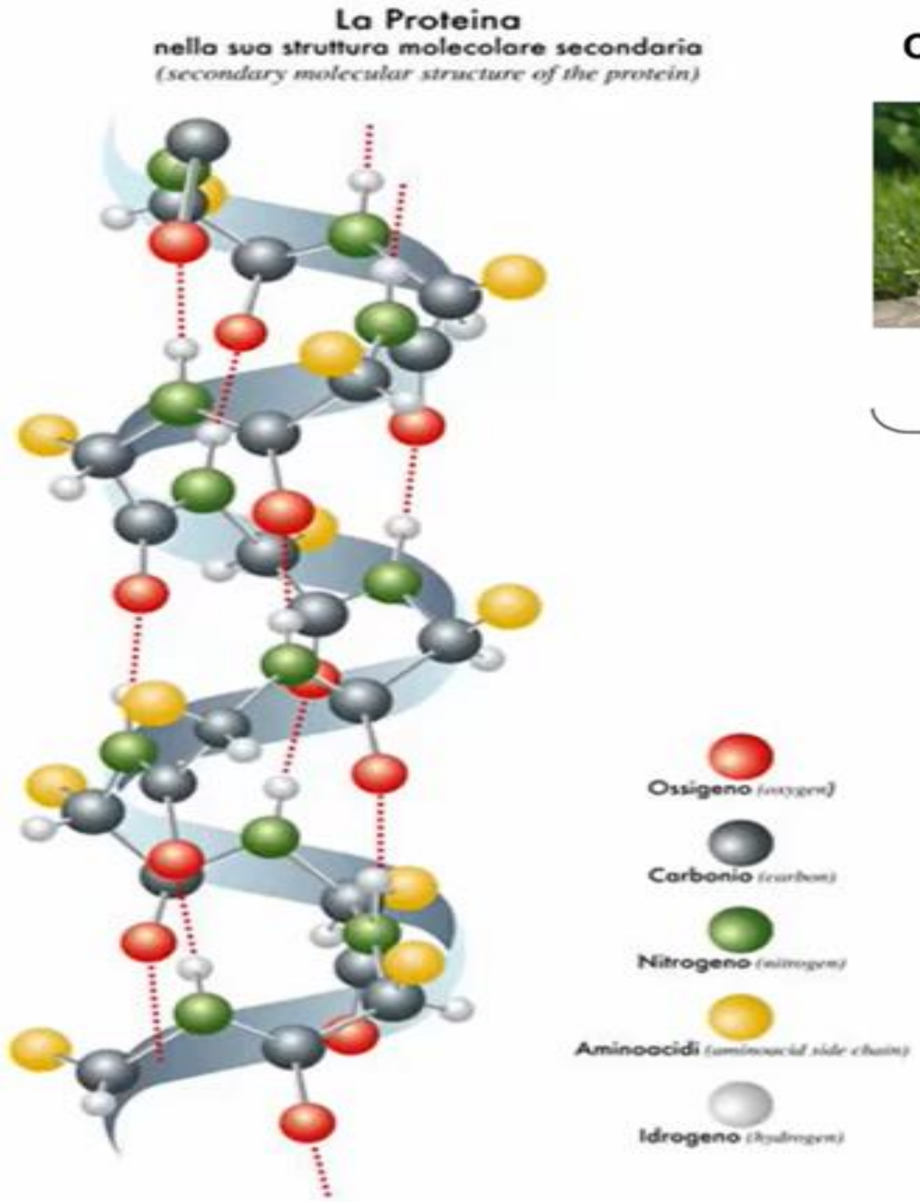
- A new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using support vectors (“essential” training tuples) and margins (defined by the support vectors)

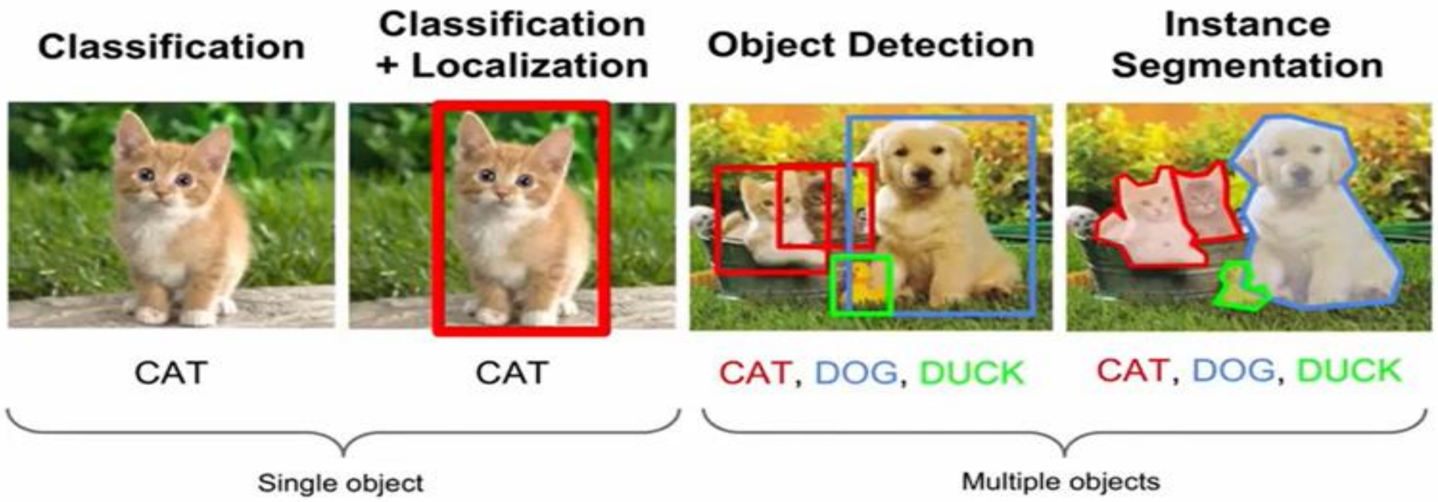
Text Classification using SVM

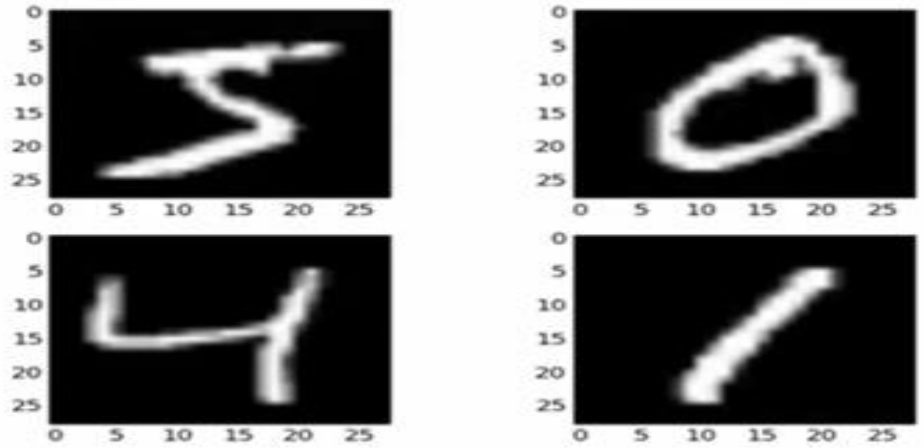



Stenography Detection
in Digital Images


Relationship
"Dog"
Relationship
"Cat"


La Proteina
nella sua struttura molecolare secondaria
(secondary molecular structure of the protein)
Ossigeno
Carbonio
Nitrogeno
Aminoacidi
Idrogeno


Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects

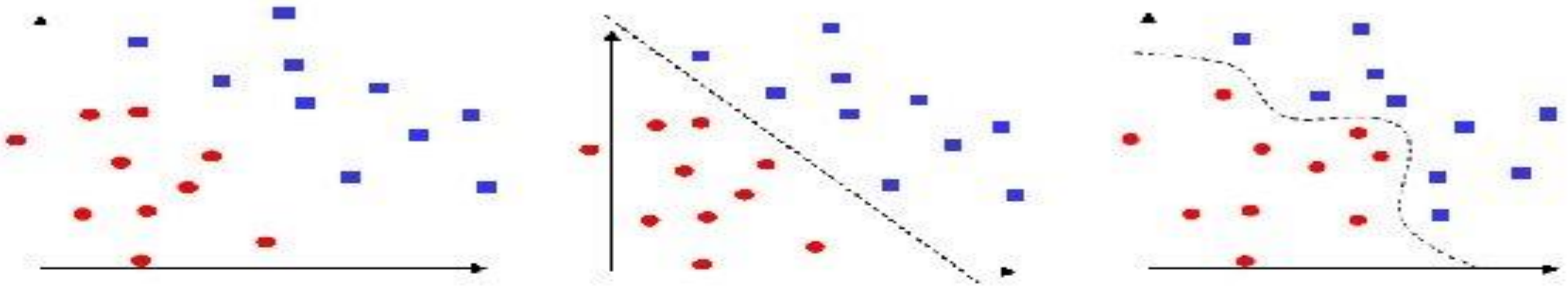


Support Vector Machine (SVM)

-- classifier, forward neural network, supervised learning

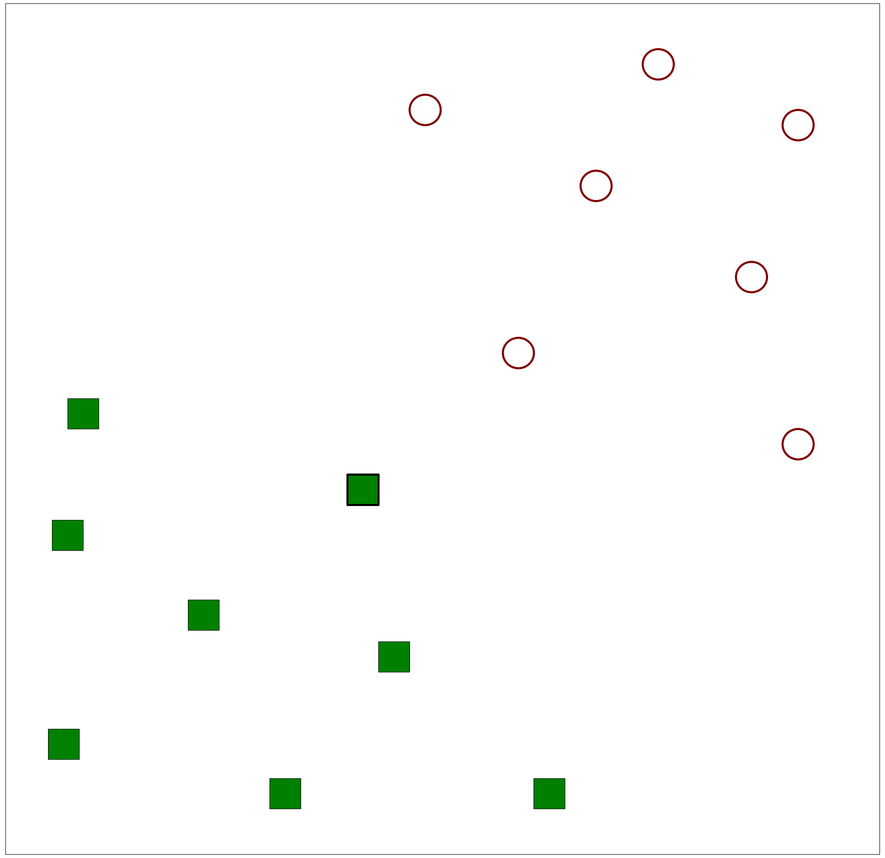
Difficulties with SVM:

i) binary classifier, ii) linearly separable patterns



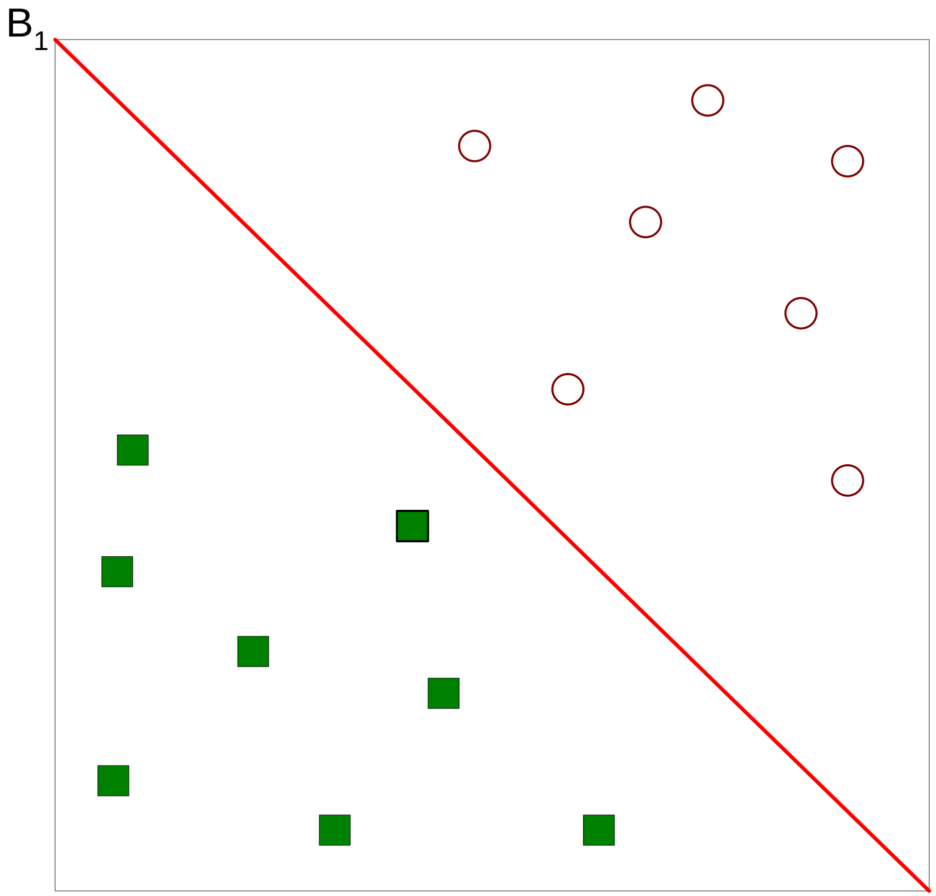
1

Support Vector Machines



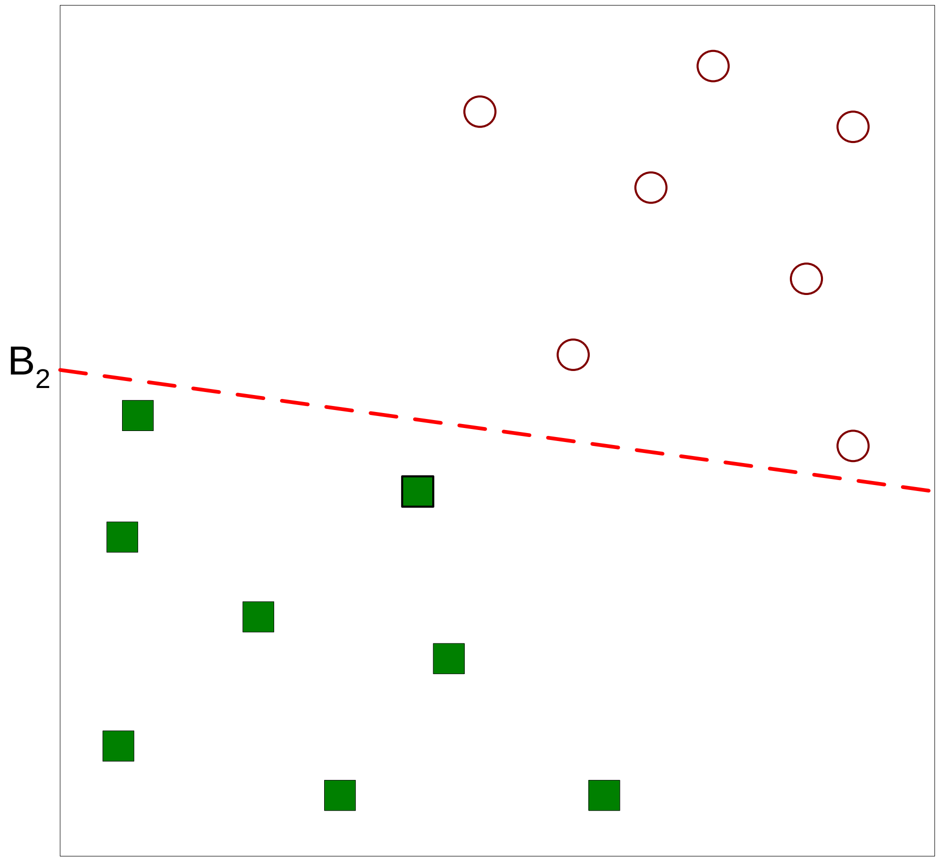
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



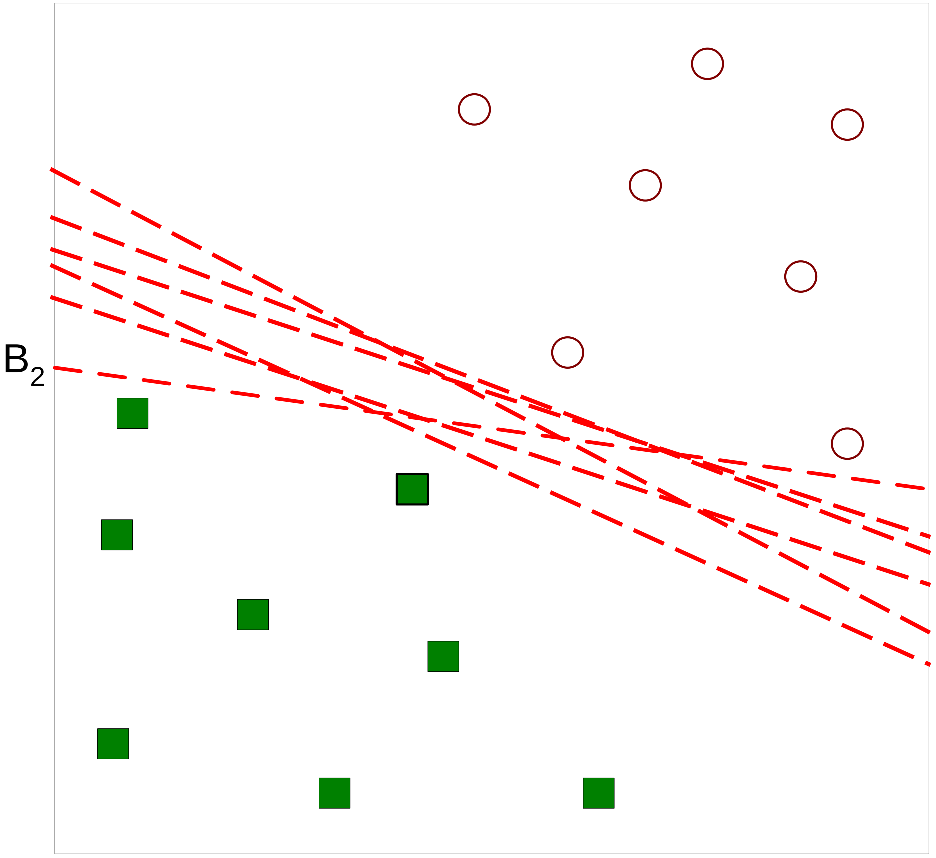
- One Possible Solution

Support Vector Machines



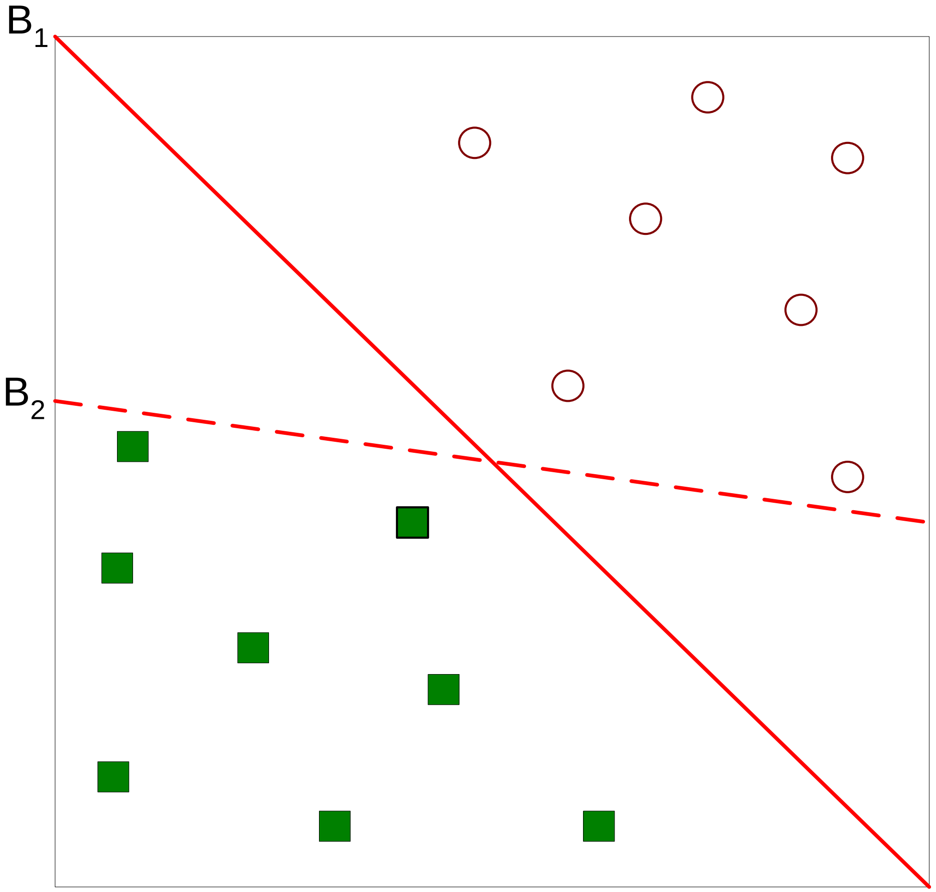
- Another possible solution

Support Vector Machines



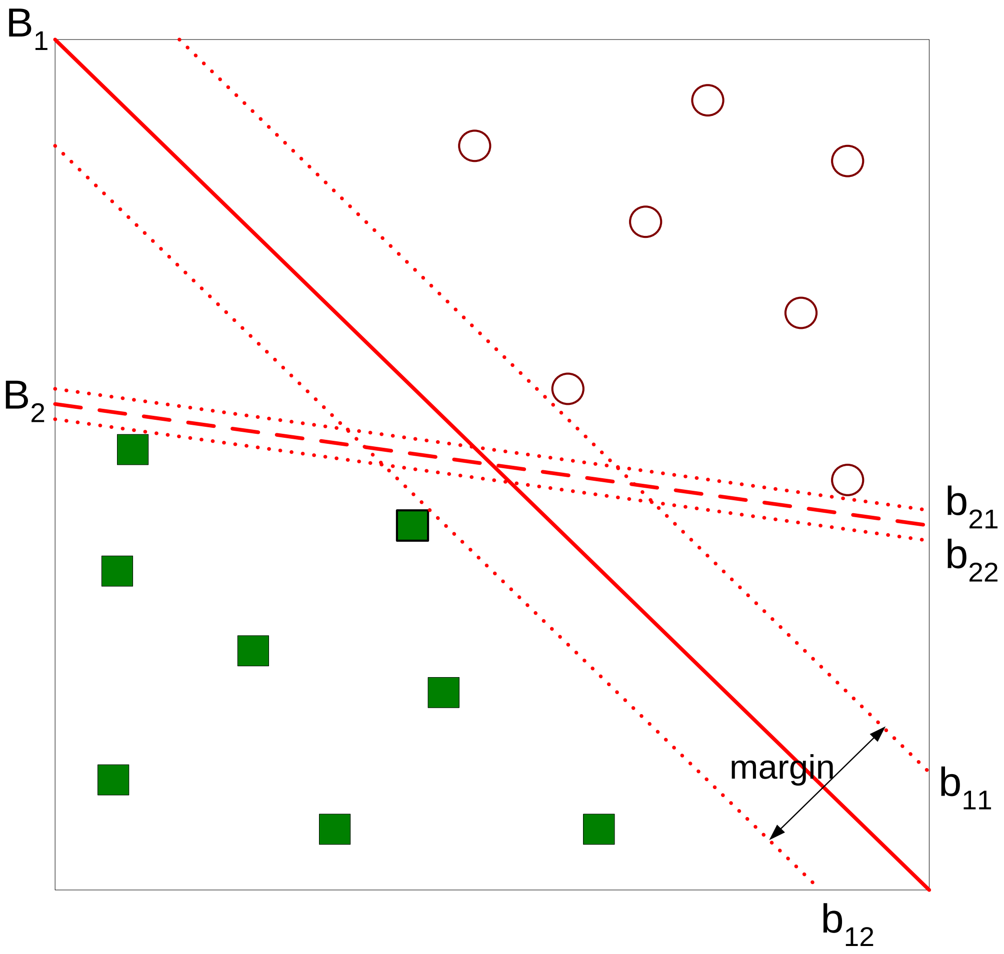
- Other possible solutions

Support Vector Machines

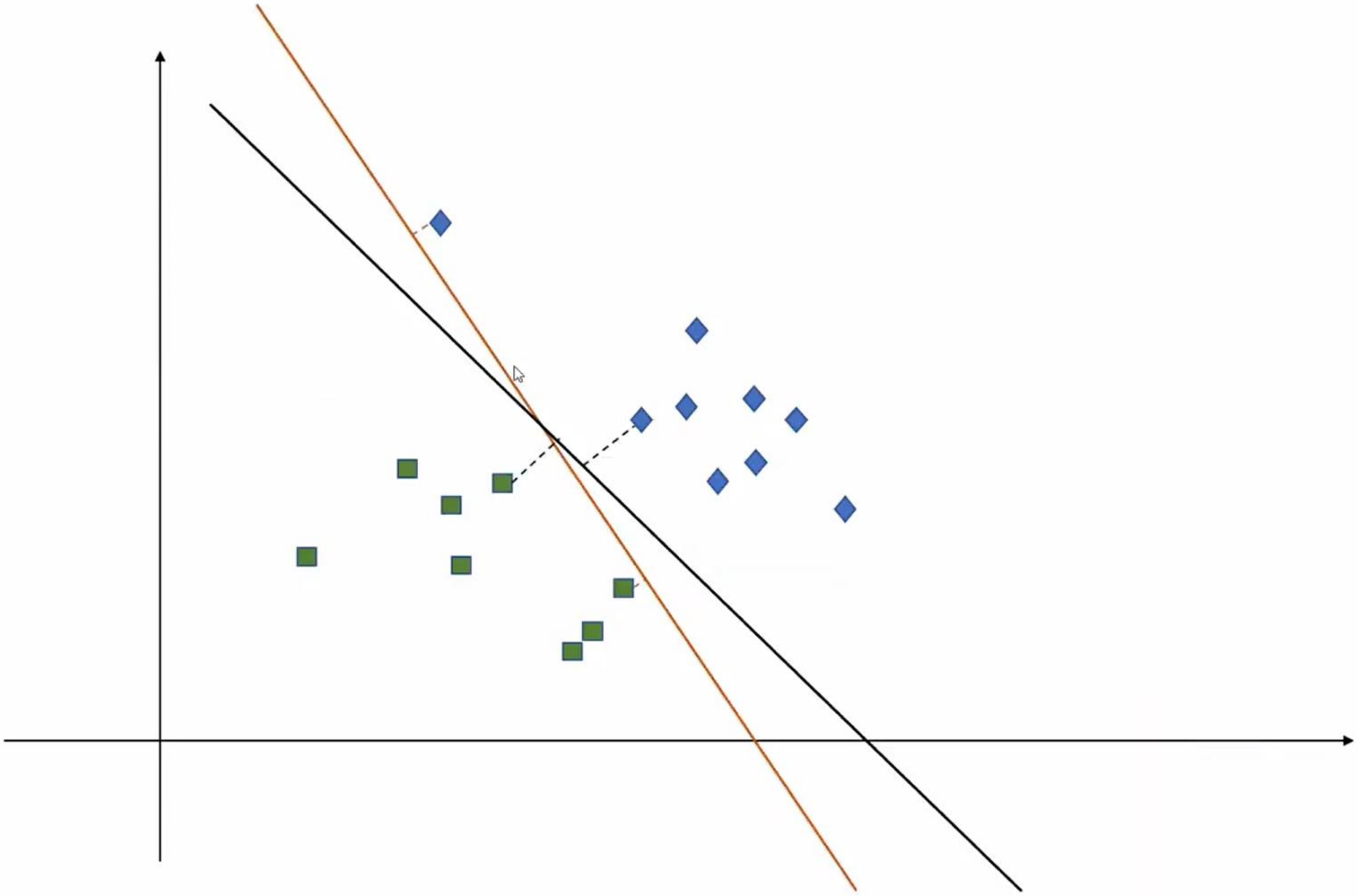


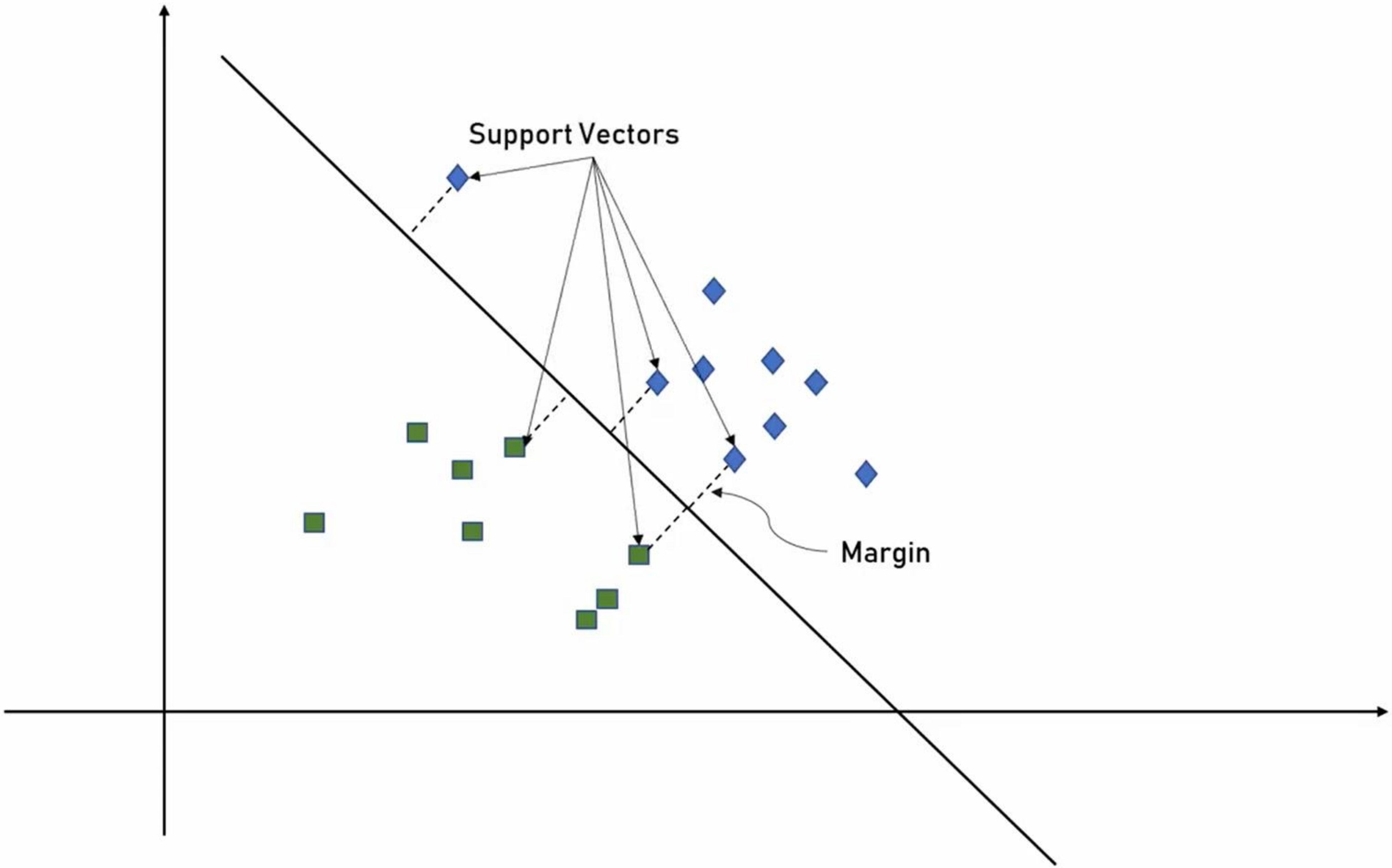
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



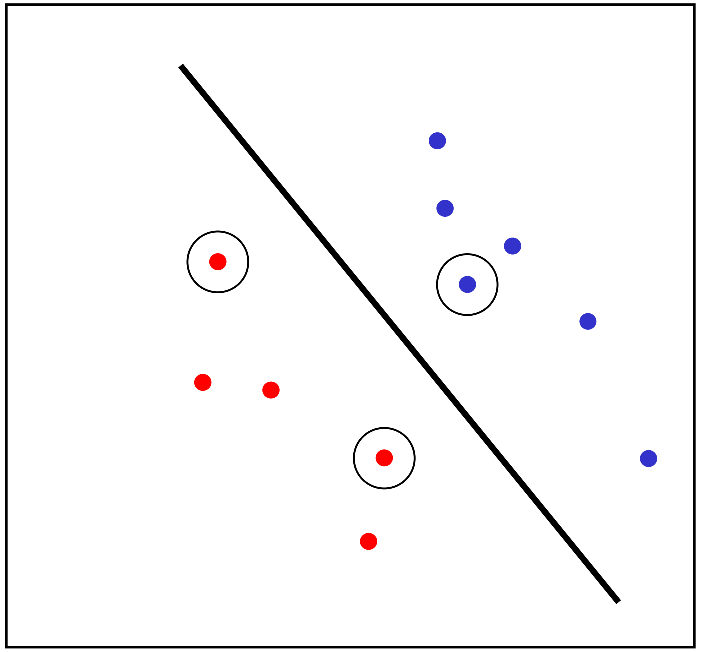
- Find hyperplane maximizes the margin => B1 is better than B2



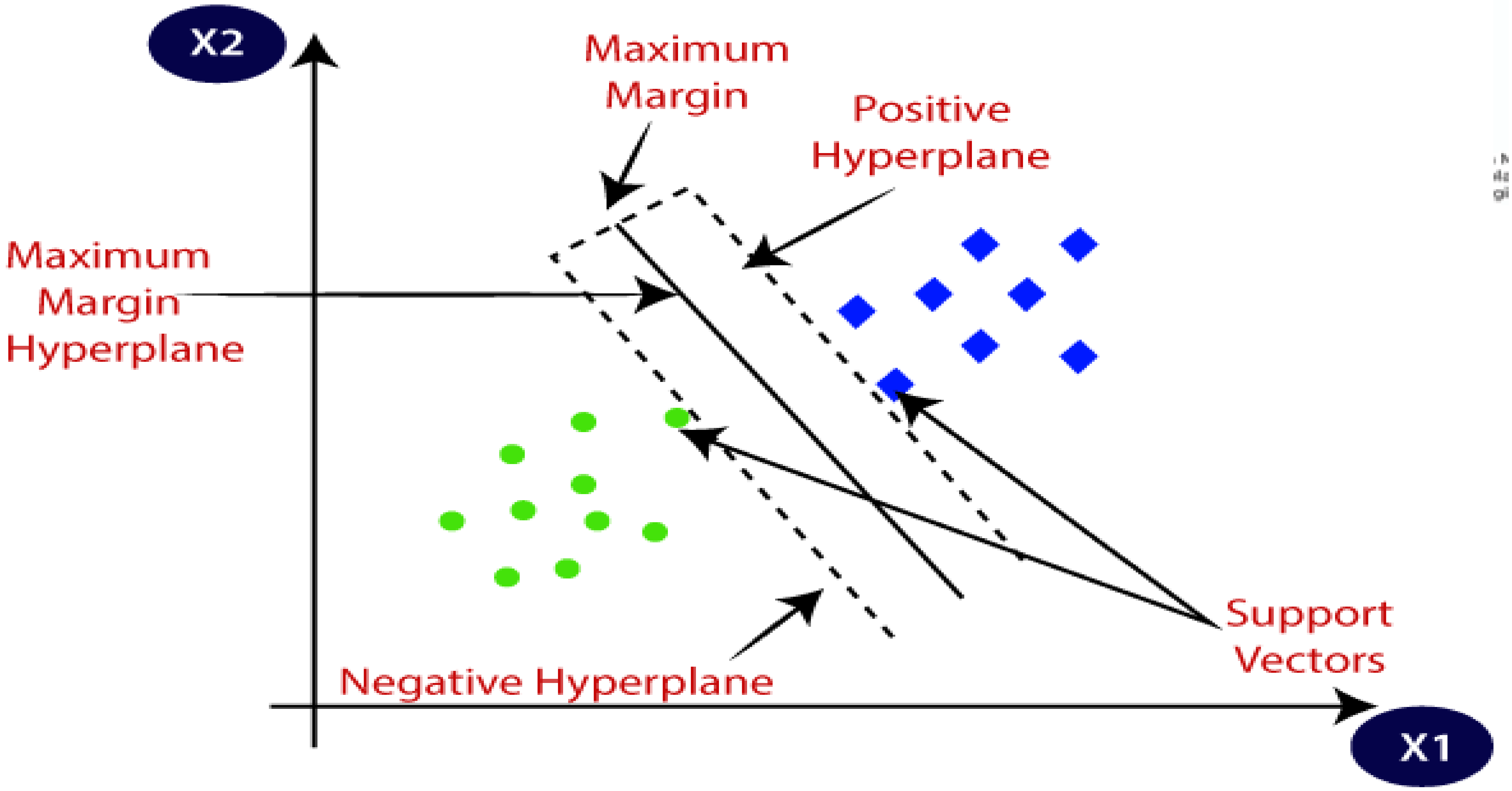

Support Vectors
Margin

Support Vector Machines

- The line that maximizes the minimum margin is a good bet.
  - The model class of “hyper-planes with a margin of m” has a low VC dimension if m is big.
- This maximum-margin separator is determined by a subset of the datapoints.
  - Datapoints in this subset are called “support vectors”.
  - It will be useful computationally if only a small fraction of the datapoints are support vectors, because we use the support vectors to decide which side of the separator a test case is on.



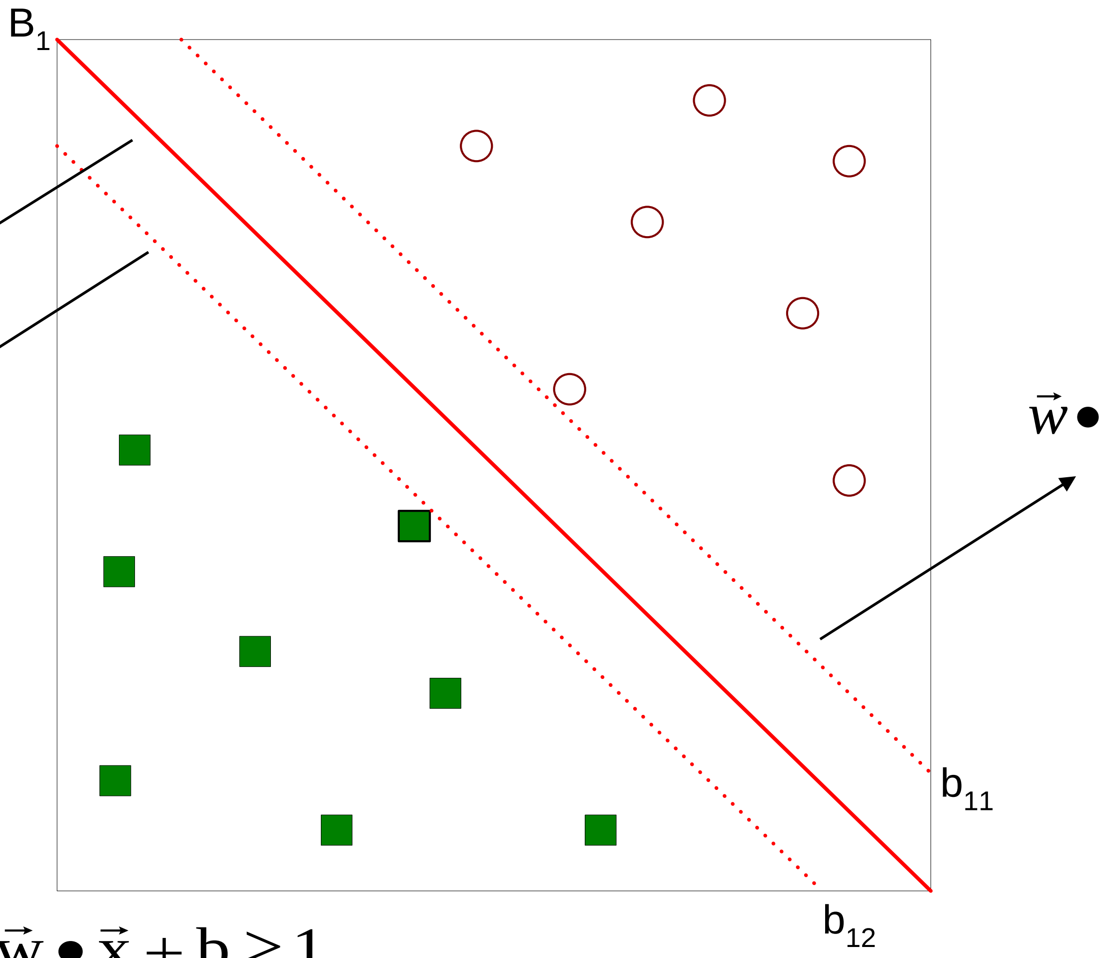
The support vectors are indicated by the circles around them.


X2
Maximum
Margin
Positive
Hyperplane
Maximum
Margin
Hyperplane
Negative Hyperplane
Support
Vectors
X1

Support Vector Machines

$$\vec{w} \bullet \vec{x} + b = 0$$

$$\vec{w} \bullet \vec{x} + b = -1$$



$$\vec{w} \bullet \vec{x} + b = +1$$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

Training a linear SVM

- To find the maximum margin separator, we have to solve the following optimization problem:

$$\mathbf{w}.\mathbf{x}^c + b > +1 \quad \textit{for positive cases}$$

$$\mathbf{w}.\mathbf{x}^c + b < -1 \quad \textit{for negative cases}$$

$$\textit{and} \quad \|\mathbf{w}\|^2 \;\; \textit{is as small as possible}$$

- This is tricky but it’s a convex problem. There is only one optimum and we can find it without fiddling with learning rates or weight decay or early stopping.
  - Don’t worry about the optimization problem. It has been solved. Its called quadratic programming.
  - It takes time proportional to N^2 which is really bad for very big datasets
    - so for big datasets we end up doing approximate optimization!

Testing a linear SVM

- The separator is defined as the set of points for which:

$$\mathbf{w}.\mathbf{x} + b = 0$$

$$so\ if\ \ \mathbf{w}.\mathbf{x}^c + b > 0\ \ say\ its\ a\ \ positive\ \ case$$

$$and\ if\ \ \mathbf{w}.\mathbf{x}^c + b < 0\ \ say\ its\ a\ negative\ \ case$$

Types of Kernel Functions

Some commonly used kernels

Polynomial: $K(\mathbf{x},\mathbf{y}) = (\mathbf{x}.\mathbf{y}+1)^p$

Gaussian radial basis function $K(\mathbf{x},\mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$

Parameters that the user must choose

Neural net: $K(\mathbf{x},\mathbf{y}) = \tanh(k\,\mathbf{x}.\mathbf{y} - \delta)$

For the neural network kernel, there is one “hidden unit” per support vector, so the process of fitting the maximum margin hyperplane decides how many hidden units to use. Also, it may violate Mercer’s condition.

Introducing slack variables

- Slack variables are constrained to be non-negative. When they are greater than zero they allow us to cheat by putting the plane closer to the datapoint than the margin. So we need to minimize the amount of cheating. This means we have to pick a value for lamba (this sounds familiar!)

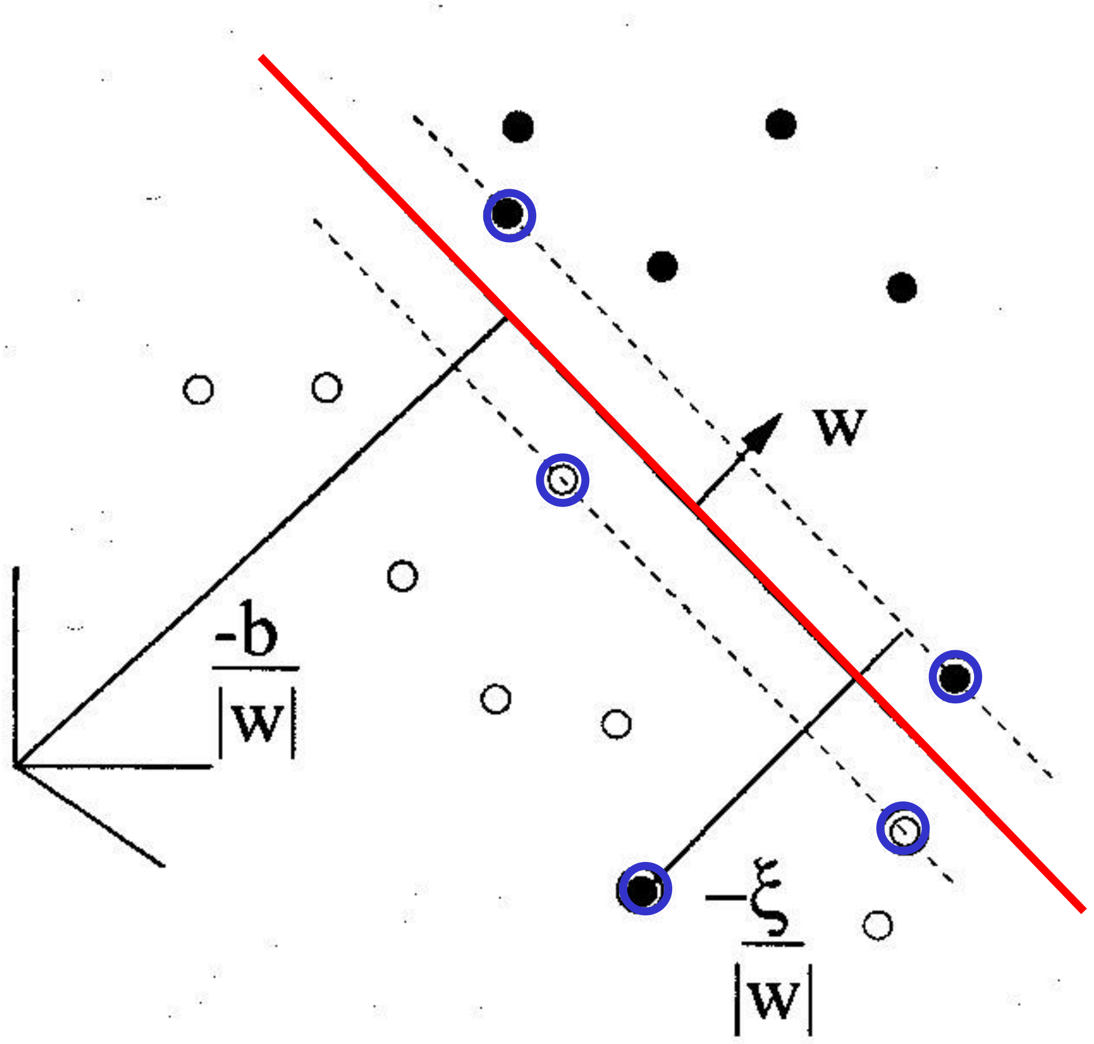
$$\mathbf{w}.\mathbf{x}^c + b \geq +1 - \xi^c \quad \textit{for positive cases}$$

$$\mathbf{w}.\mathbf{x}^c + b \leq -1 + \xi^c \quad \textit{for negative cases}$$

$$\textit{with} \quad \xi^c \geq 0 \quad \textit{for all } c$$

$$\textit{and} \quad \frac{\|\mathbf{w}\|^2}{2} + \lambda \sum_c \xi^c \quad \textit{as small as possible}$$

A picture of the best plane with a slack variable



Linear SVM

- Linear model:

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

- Learning the model is equivalent to determining the values of
  - How to find from training data?

$$\vec{w} \text{ and } b$$

$$\vec{w} \text{ and } b$$

Learning Linear SVM

- Objective is to maximize:

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

  - Which is equivalent to minimizing:

$$L(\vec{w}) = \frac{\|\vec{w}\|^2}{2}$$

  - Subject to the following constraints:

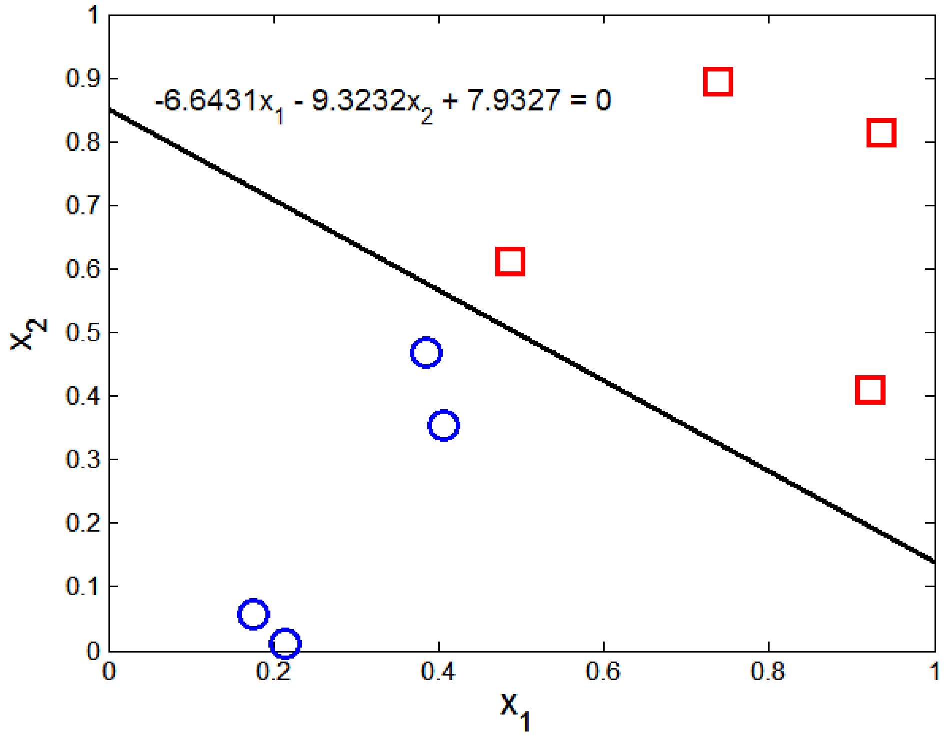
$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 \end{cases}$$

or

$$y_i(\mathrm{w} \bullet \mathrm{x}_i + b) \geq 1, \qquad i = 1,2,\ldots,N$$

- This is a constrained optimization problem
  - Solve it using Lagrange multiplier method

Example of Linear SVM



Support vectors

x1	x2	y	λ
0.3858	0.4687	1	65.5261
0.4871	0.611	-1	65.5261
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

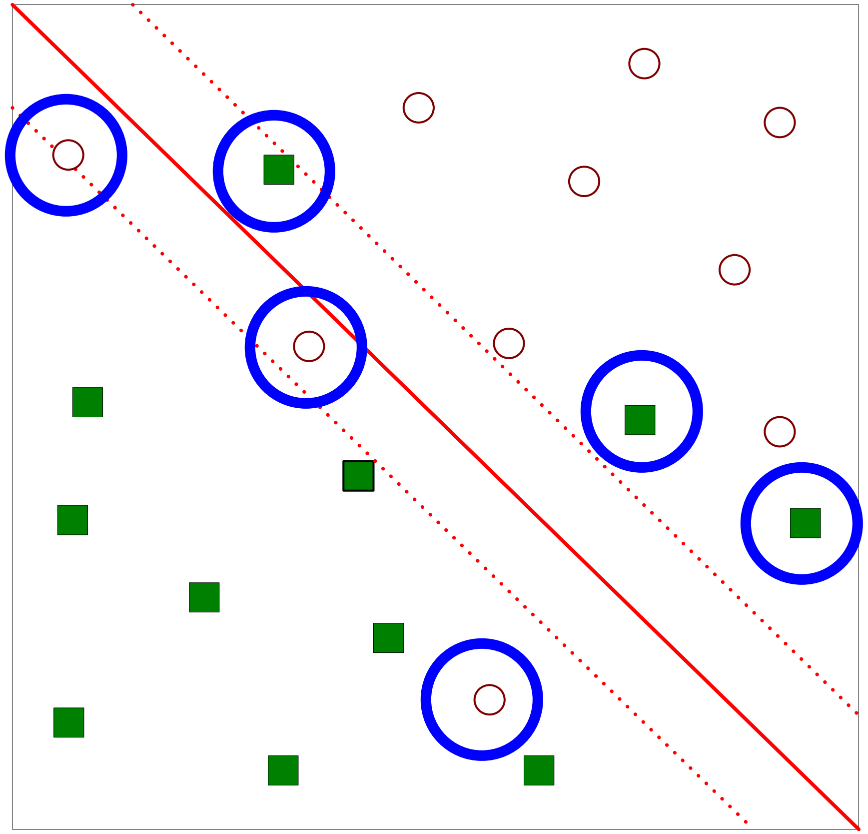
Learning Linear SVM

- Decision boundary depends only on support vectors
  - If you have data set with same support vectors, decision boundary will not change
  - How to classify using SVM once **w** and *b* are found? Given a test record, x_i

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 \end{cases}$$

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

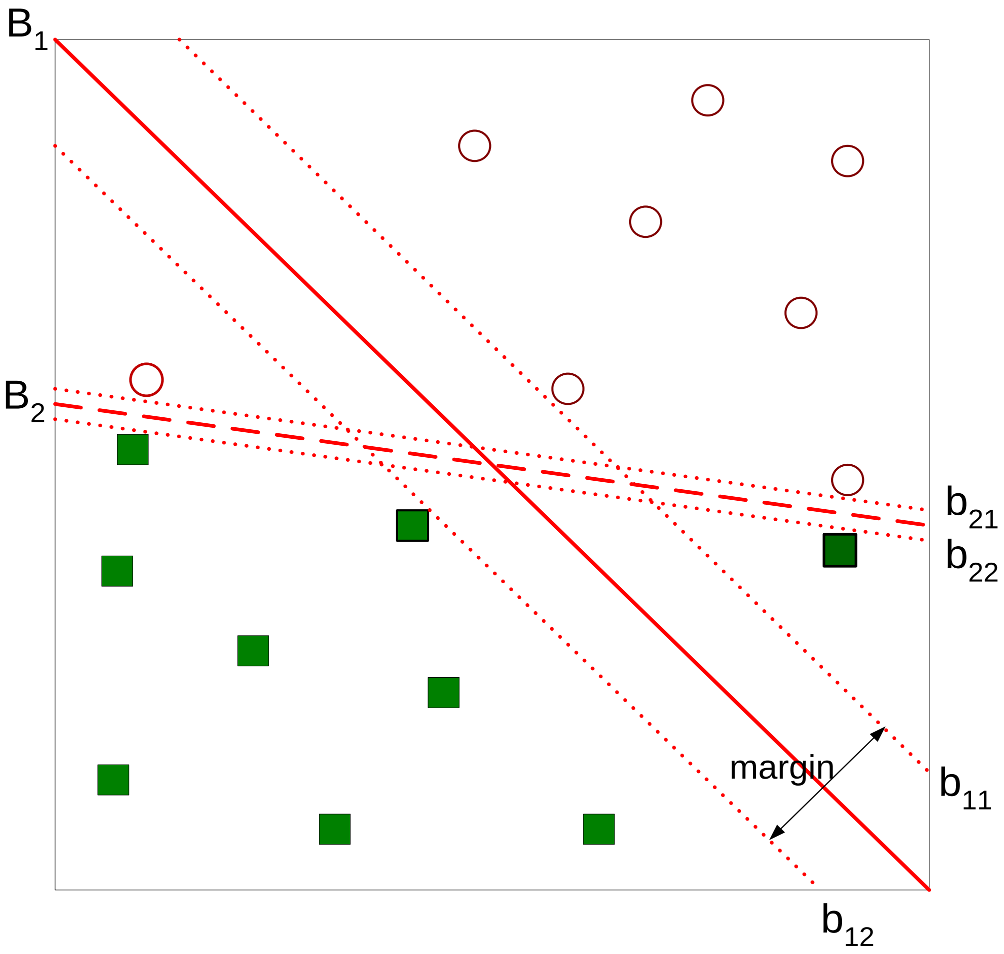
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

    - Subject to:

$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \geq 1 - \xi_\mathrm{i} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \leq -1 + \xi_\mathrm{i} \end{cases}$$

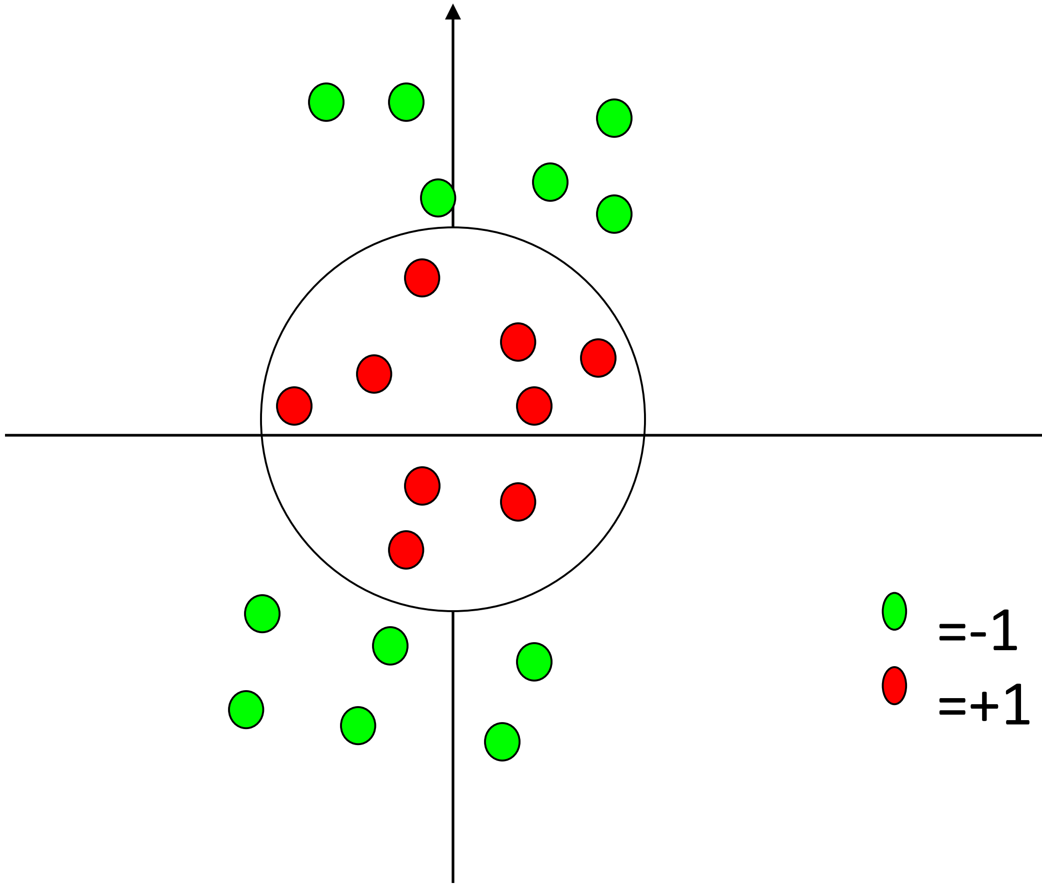
    - If k is 1 or 2, this leads to similar objective function as linear SVM but with different constraints (see textbook)

Support Vector Machines



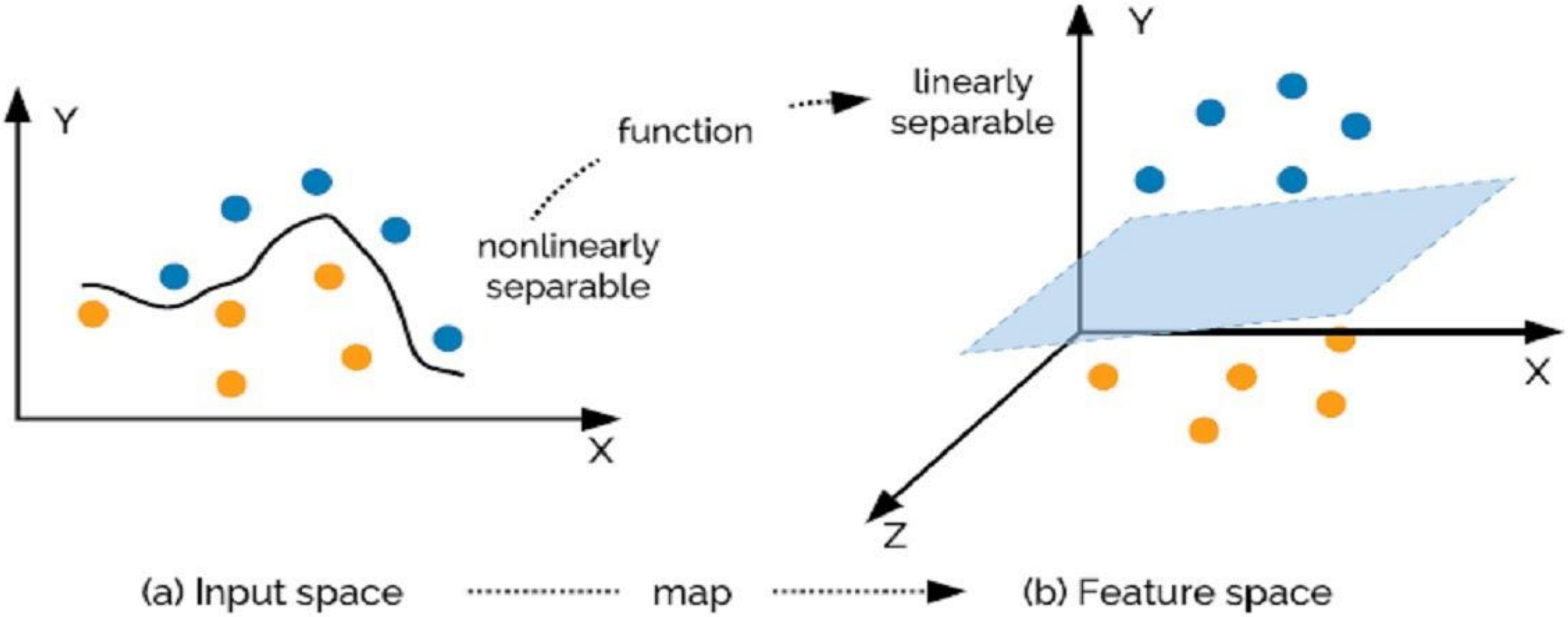
- Find the hyperplane that optimizes both factors

Problems with linear SVM



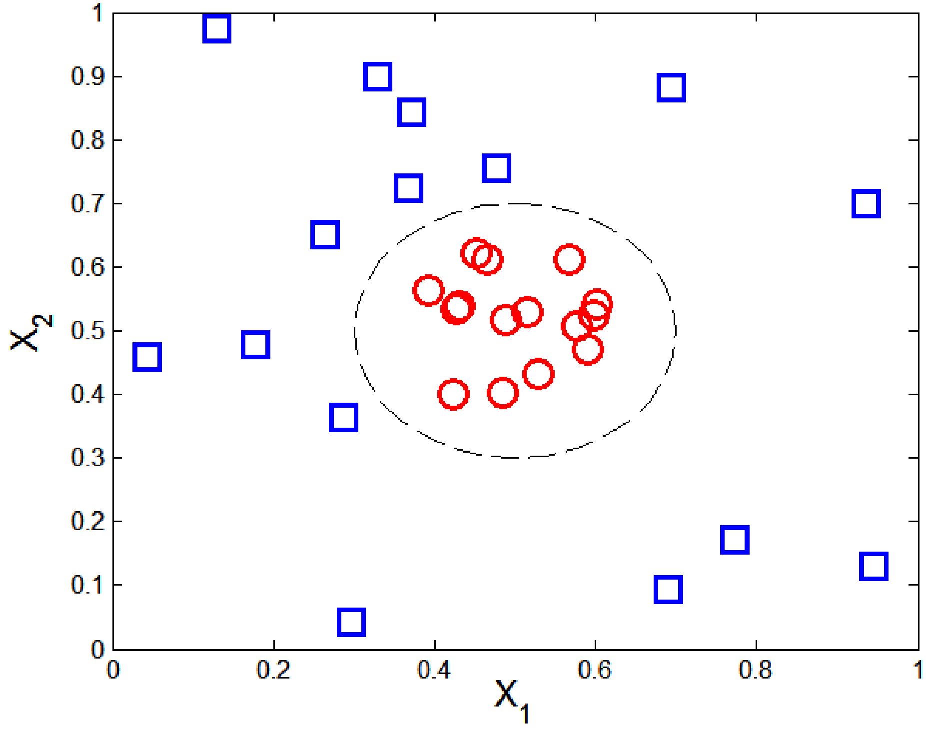
What if the decision function is not a linear?

Kernal Trick (SVM)…



Nonlinear Support Vector Machines

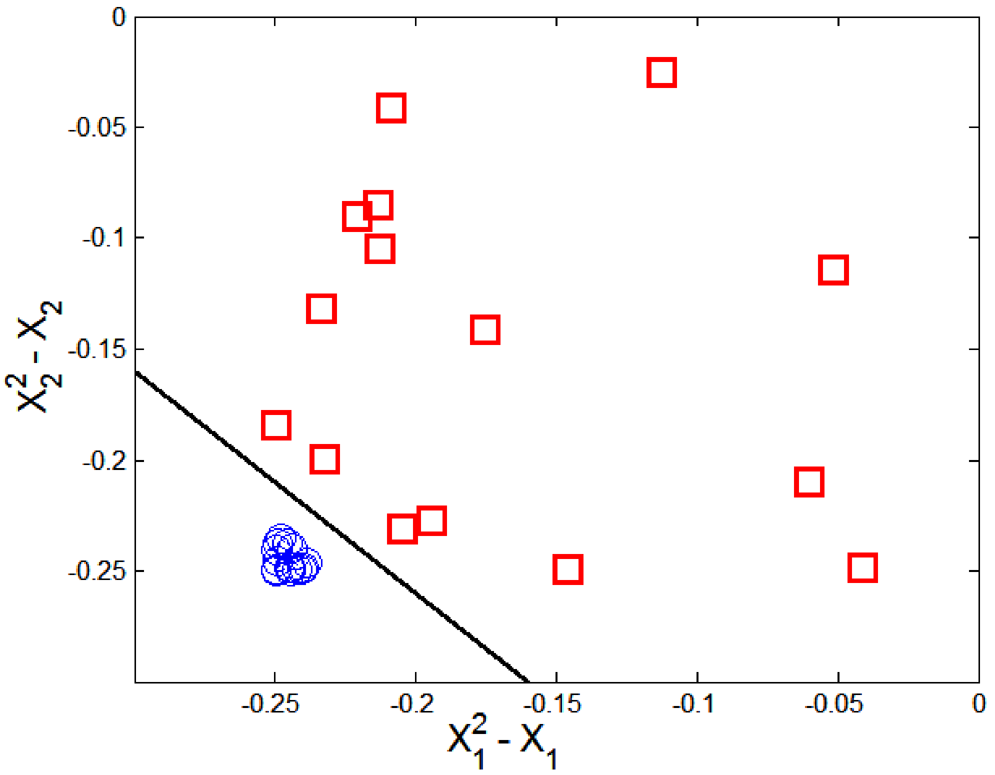
- What if decision boundary is not linear?



$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & \text{otherwise} \end{cases}$$

Nonlinear Support Vector Machines

- Transform data into higher dimensional space



$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46.$$

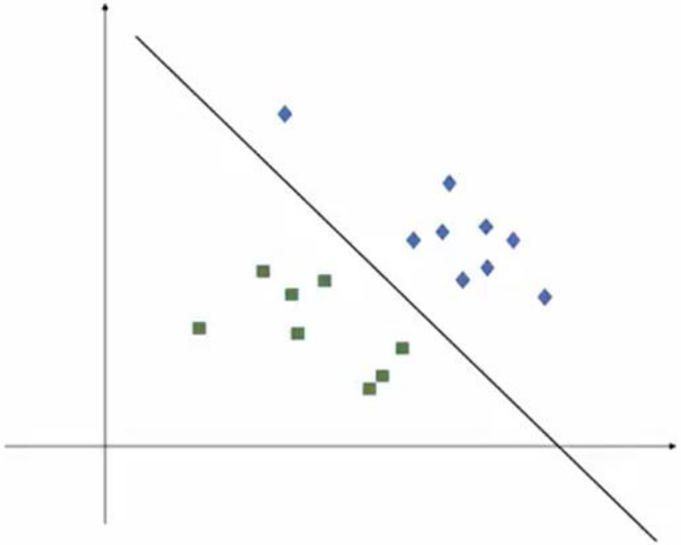
$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1).$$

$$w_4 x_1^2 + w_3 x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

Decision boundary:

$$\vec{w} \bullet \Phi(\vec{x}) + b = 0$$

2D



3D

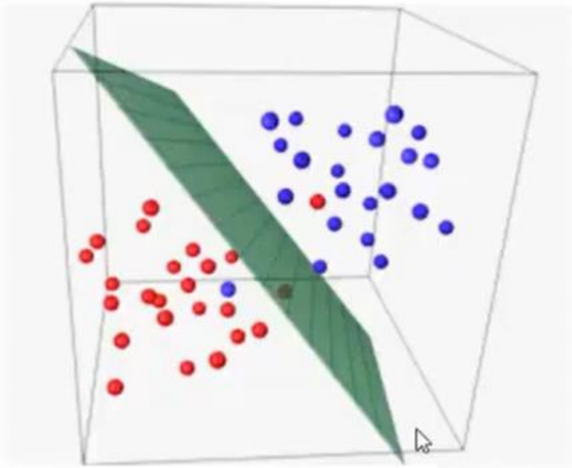
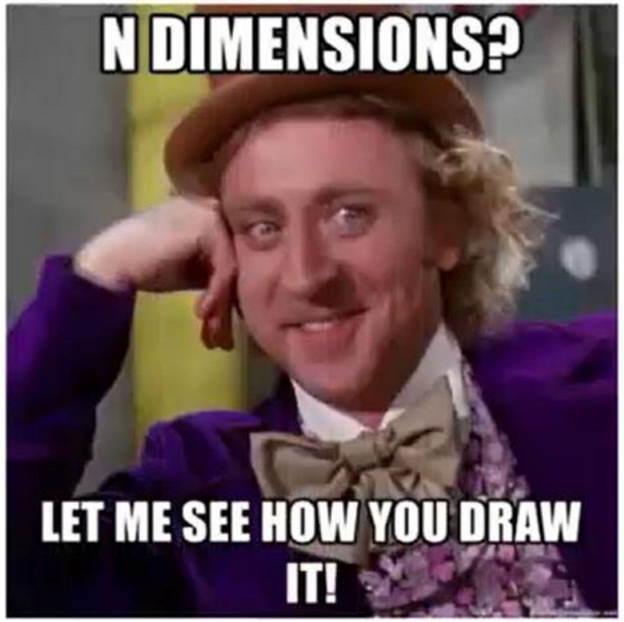
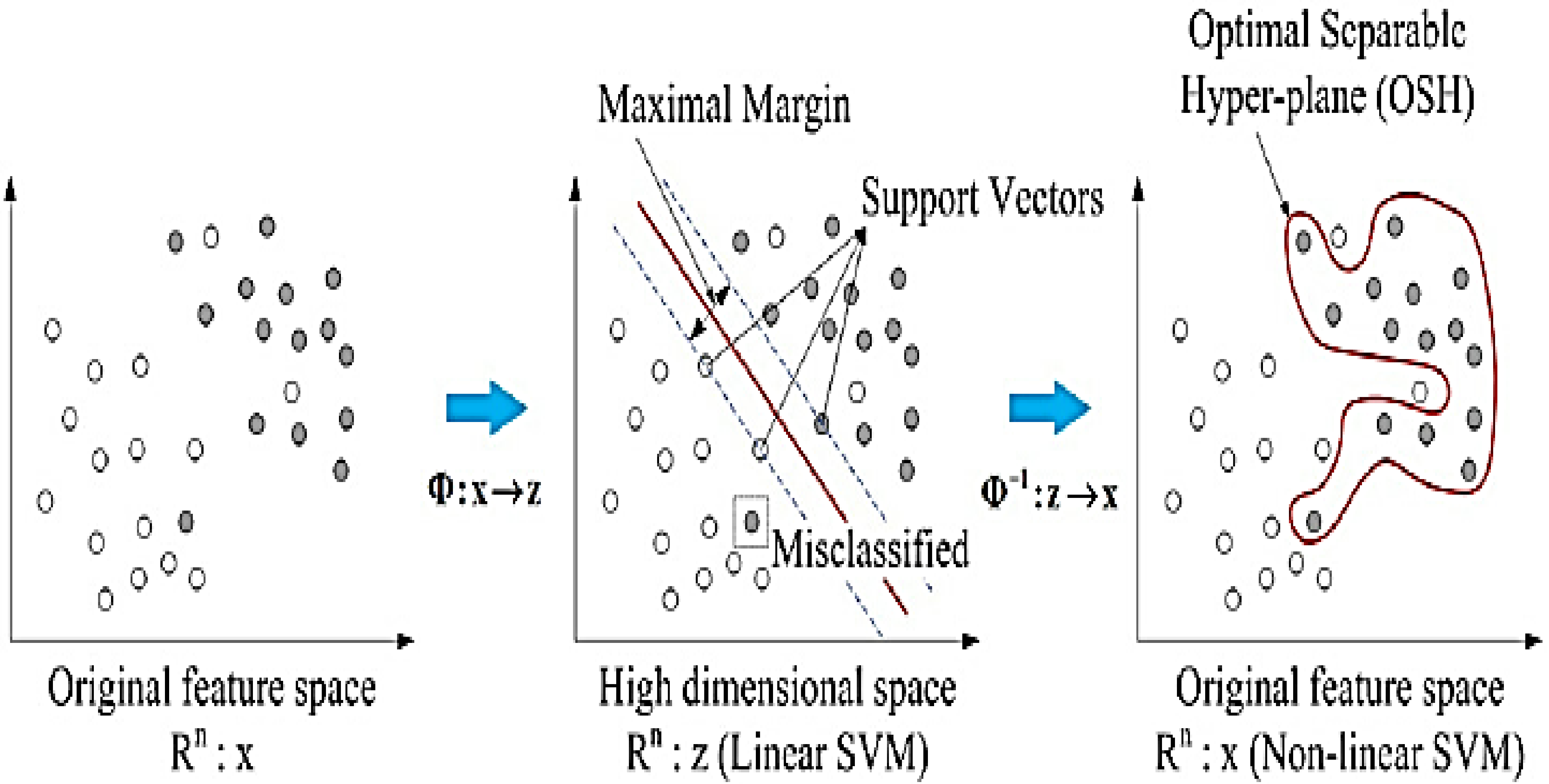


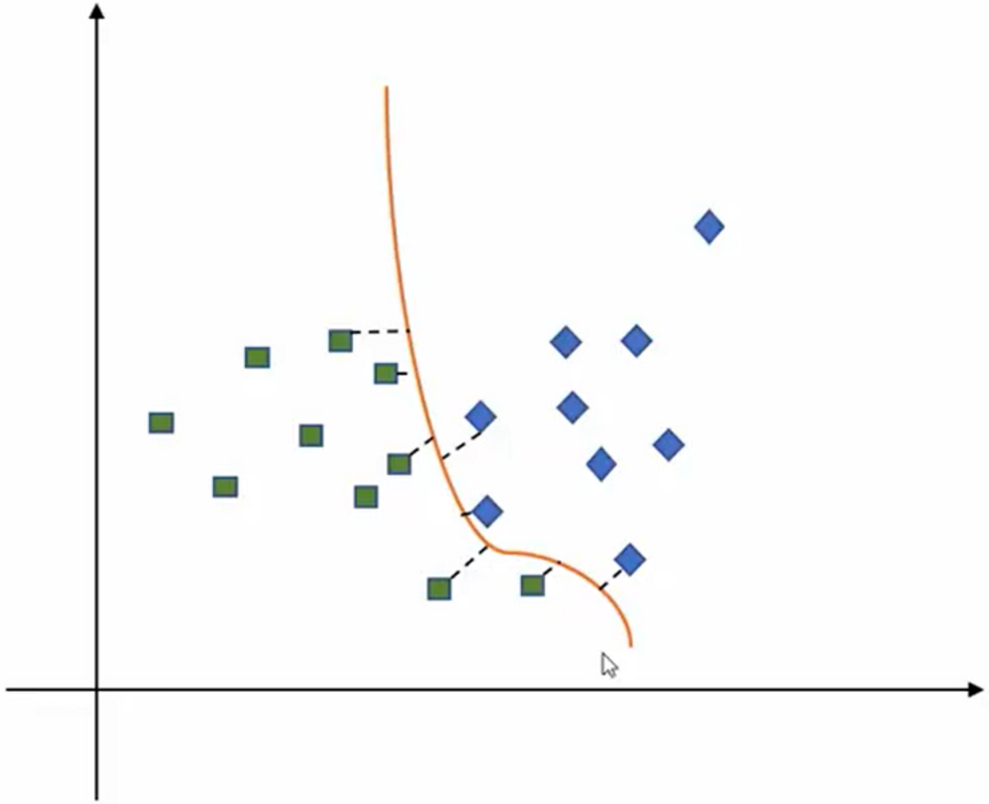
Image Credit: https://appliedmachinelearning.blog

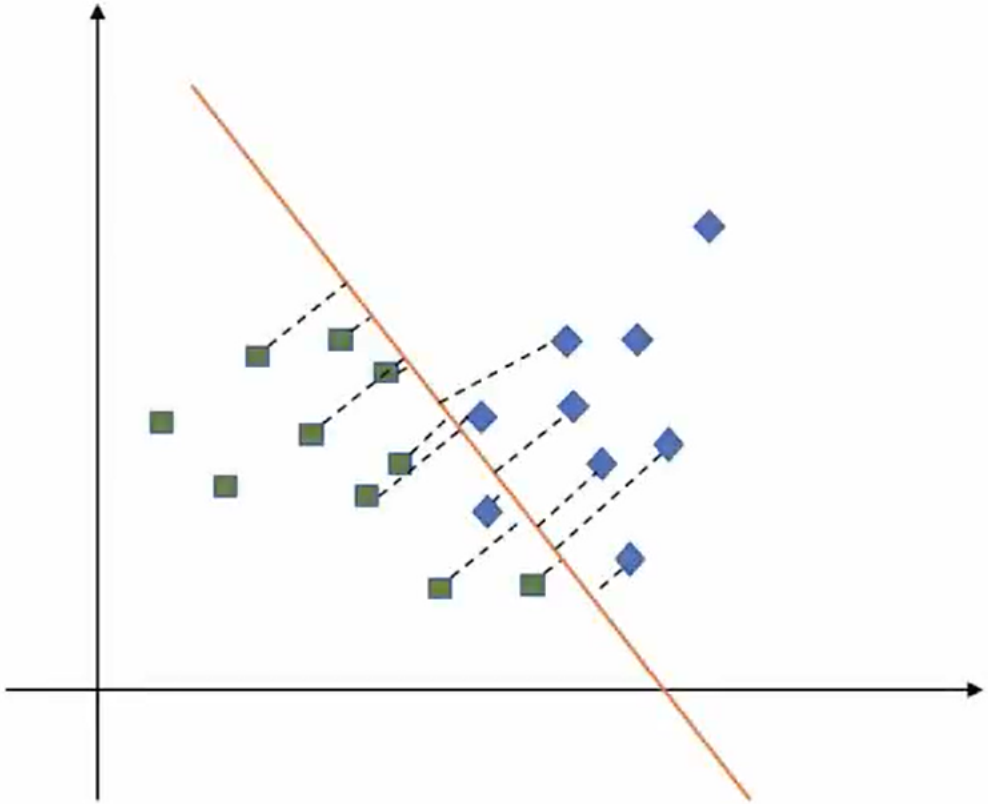
nD

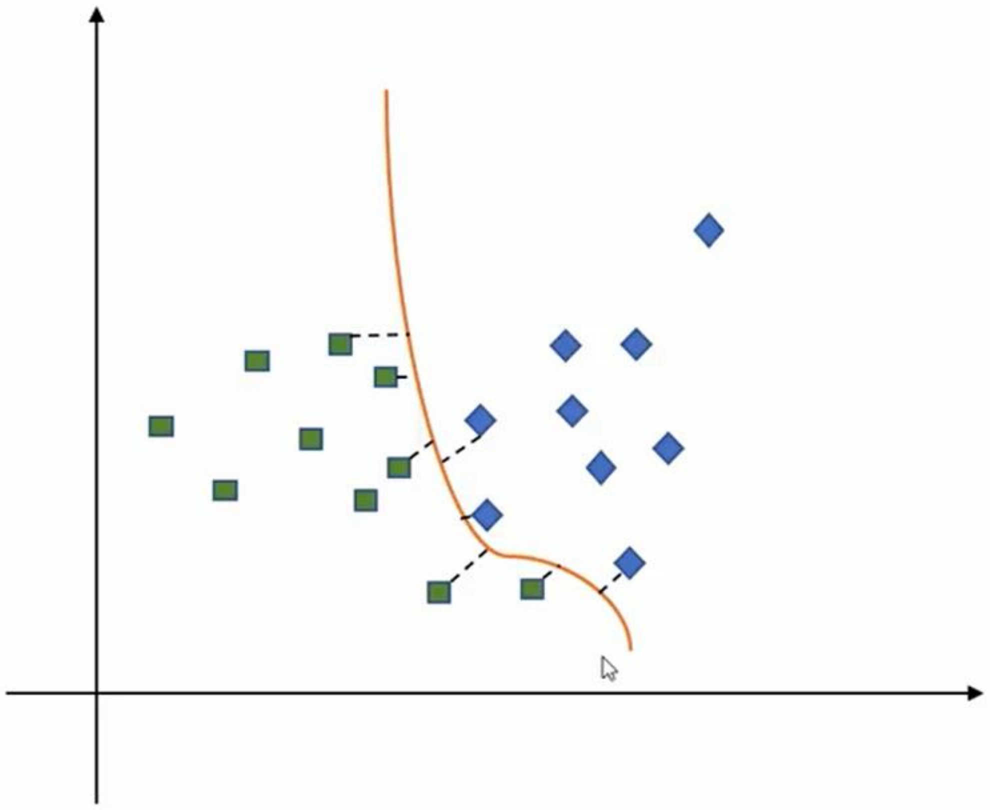



Optimal Separable
Hyper-plane (OSH)
Maximal Margin
Support Vectors
Misclassified
$\Phi: x \rightarrow z$
$\Phi^{-1}: z \rightarrow x$
Original feature space
$R^n : x$
High dimensional space
$R^n : z$ (Linear SVM)
Original feature space
$R^n : x$ (Non-linear SVM)

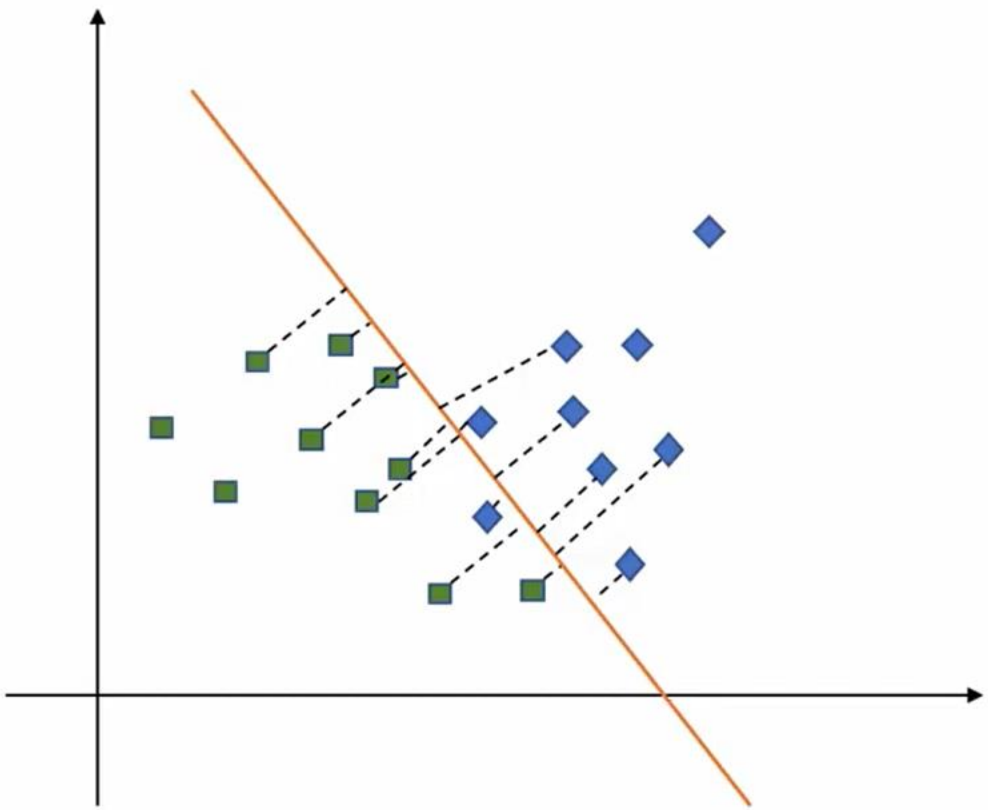
Gamma & Regularization



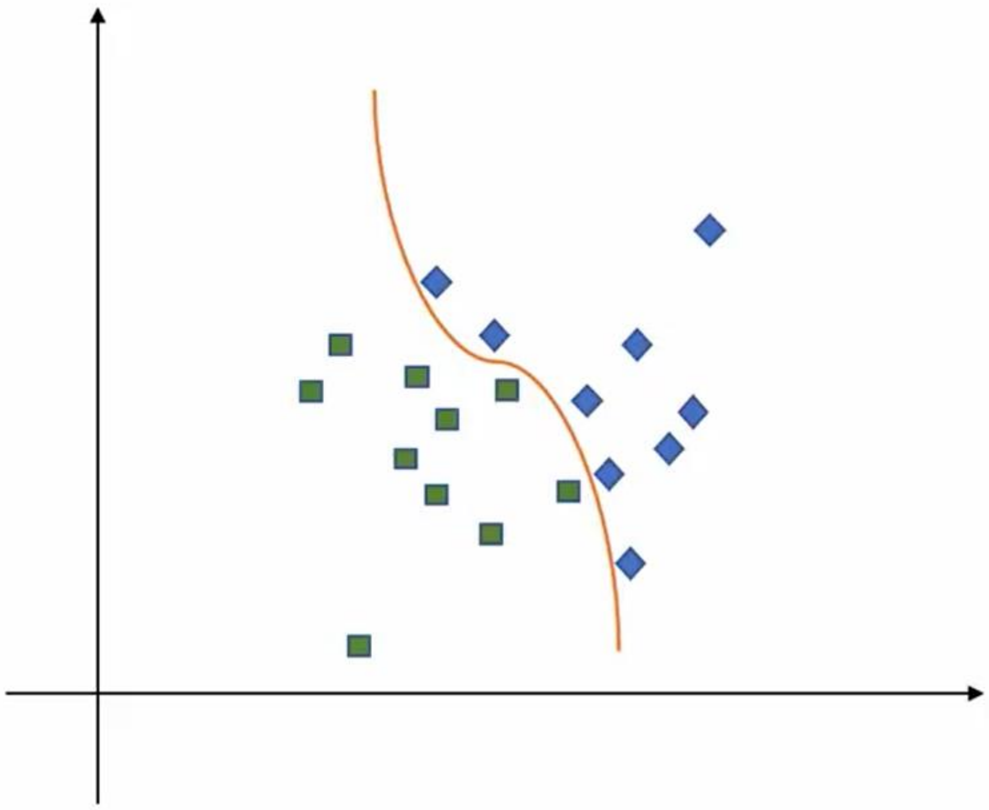




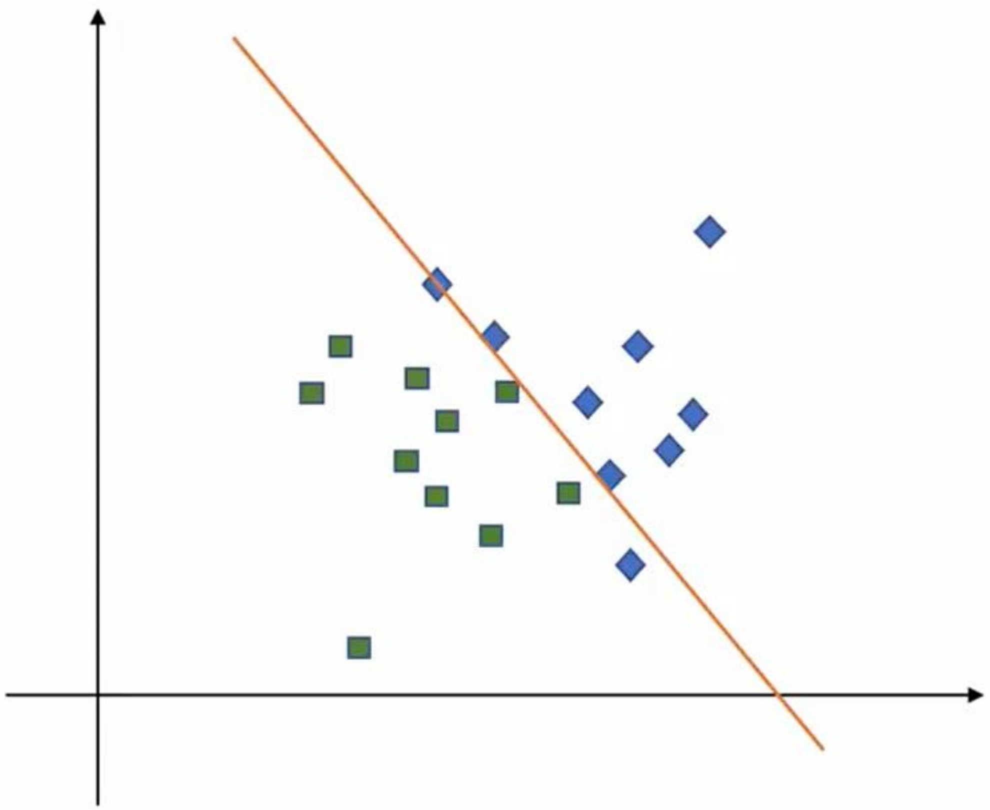
High Gamma



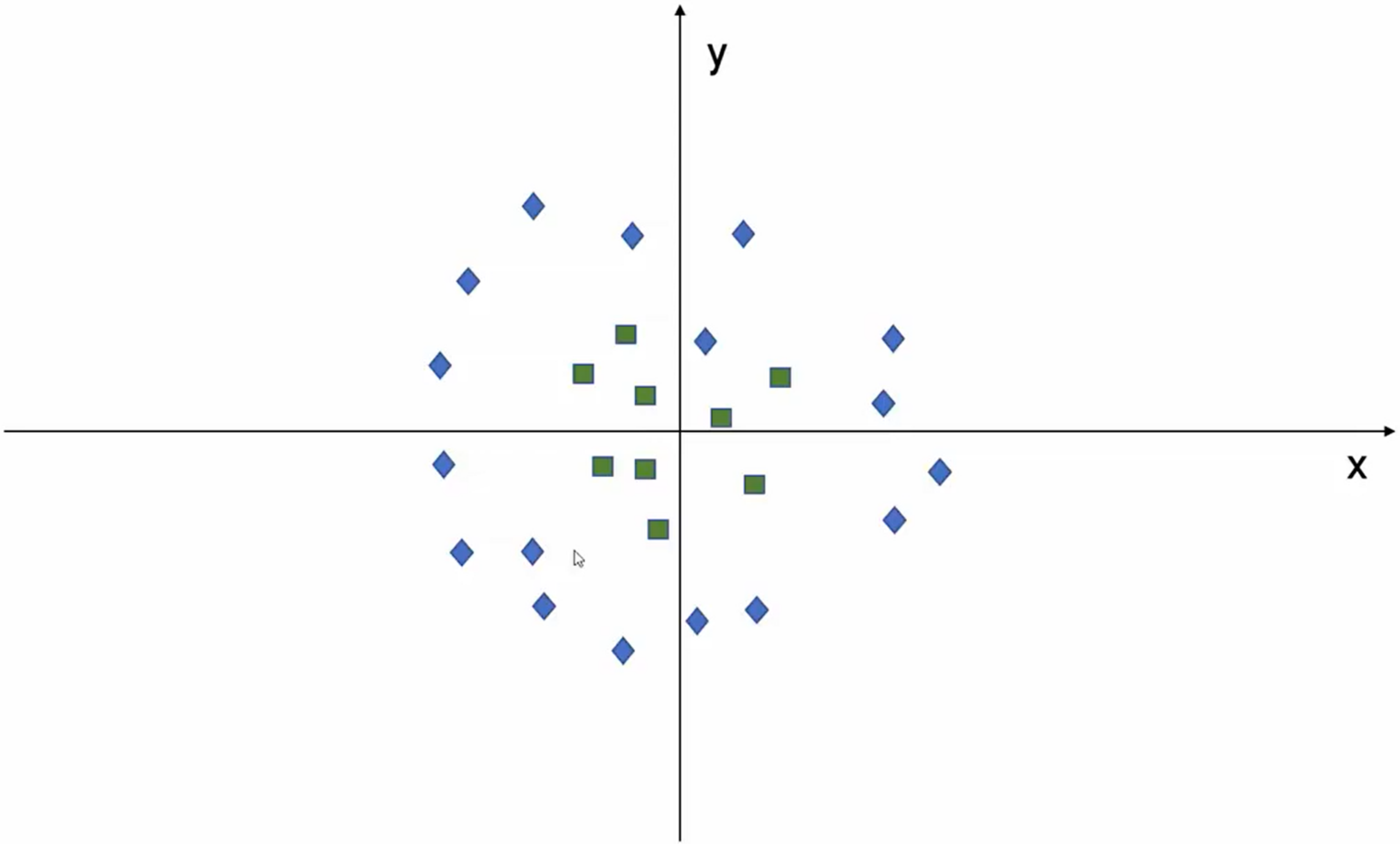
Low Gamma

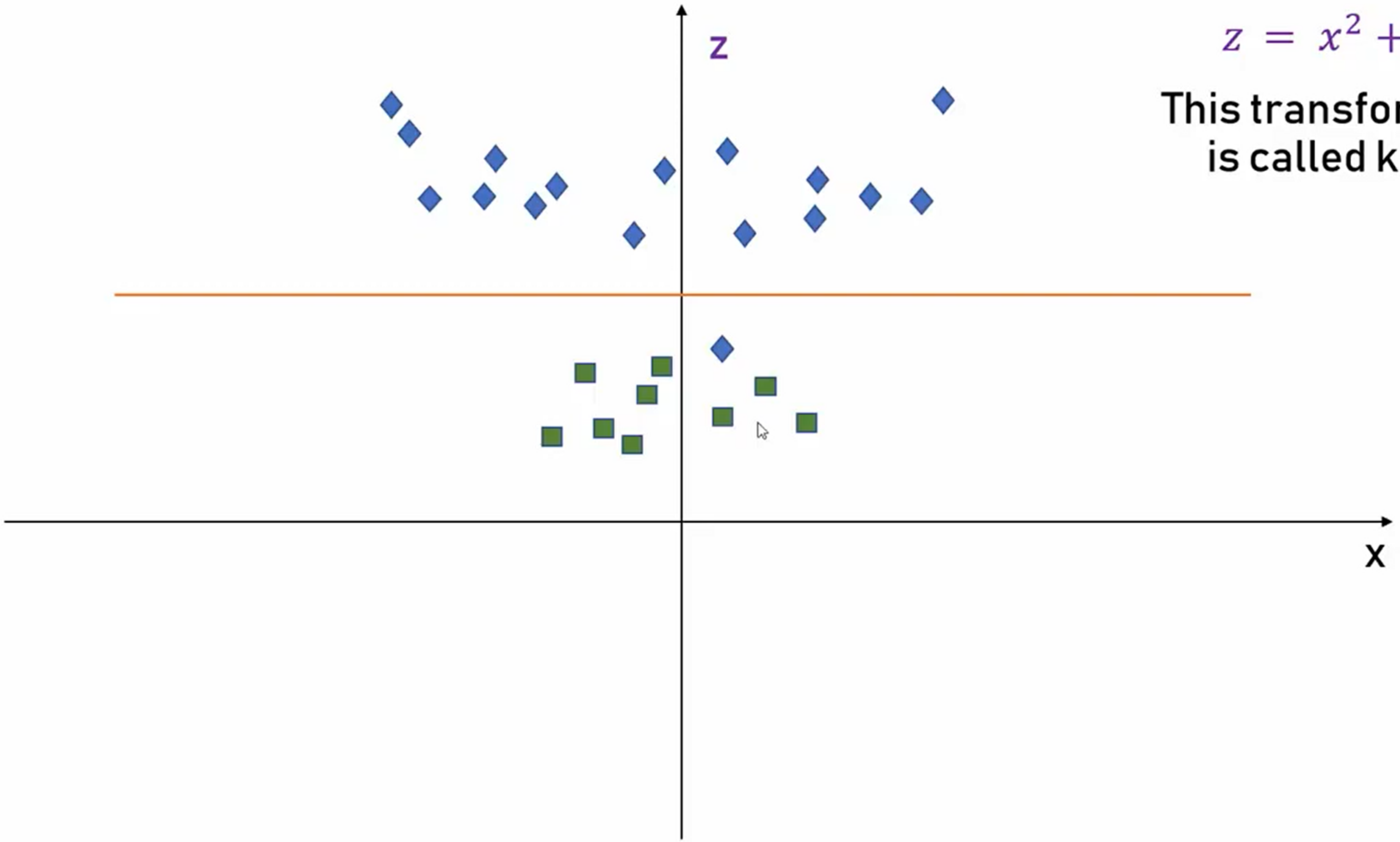


High Regularization (C)



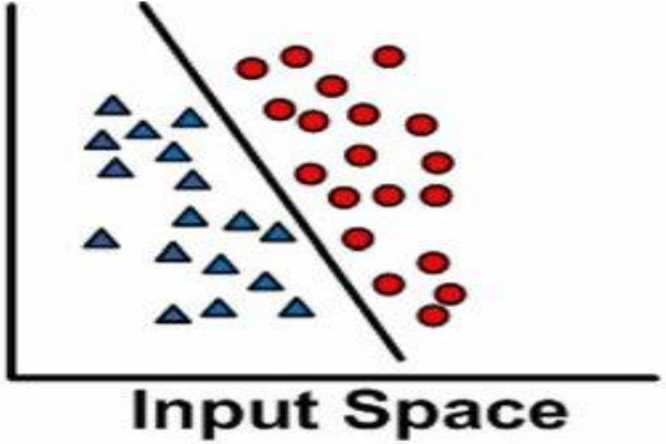
Low Regularization (C)


y
x

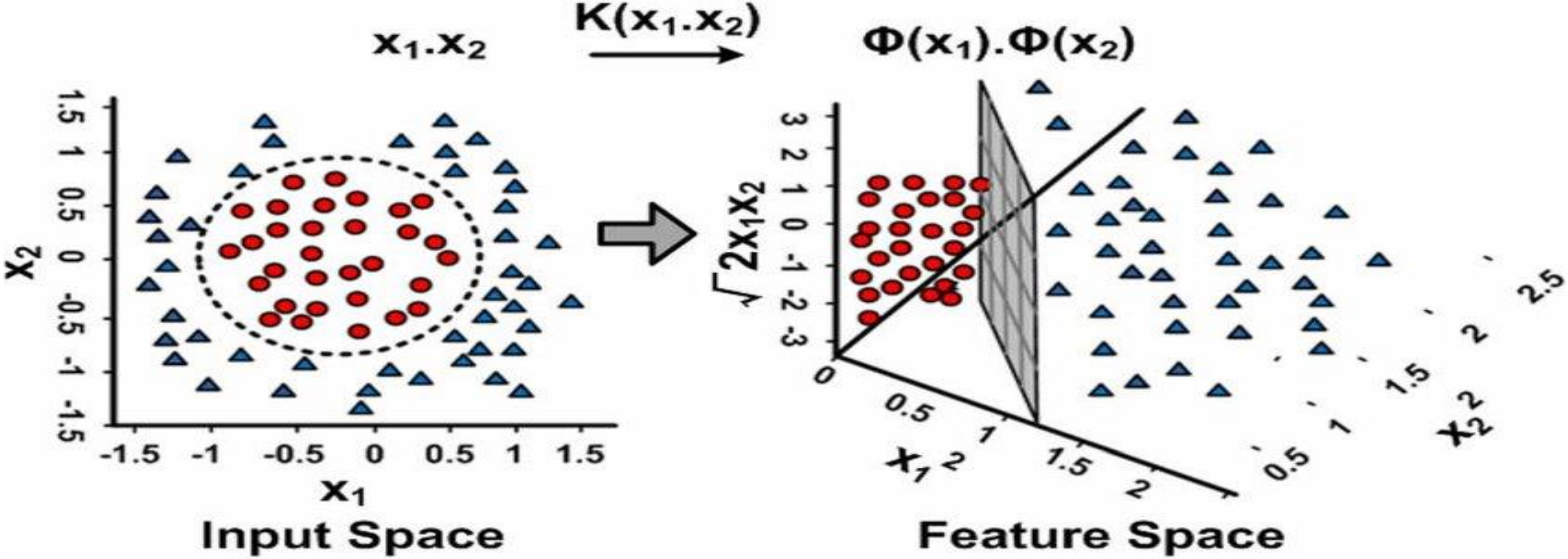


$$z = x^2 + y^2$$

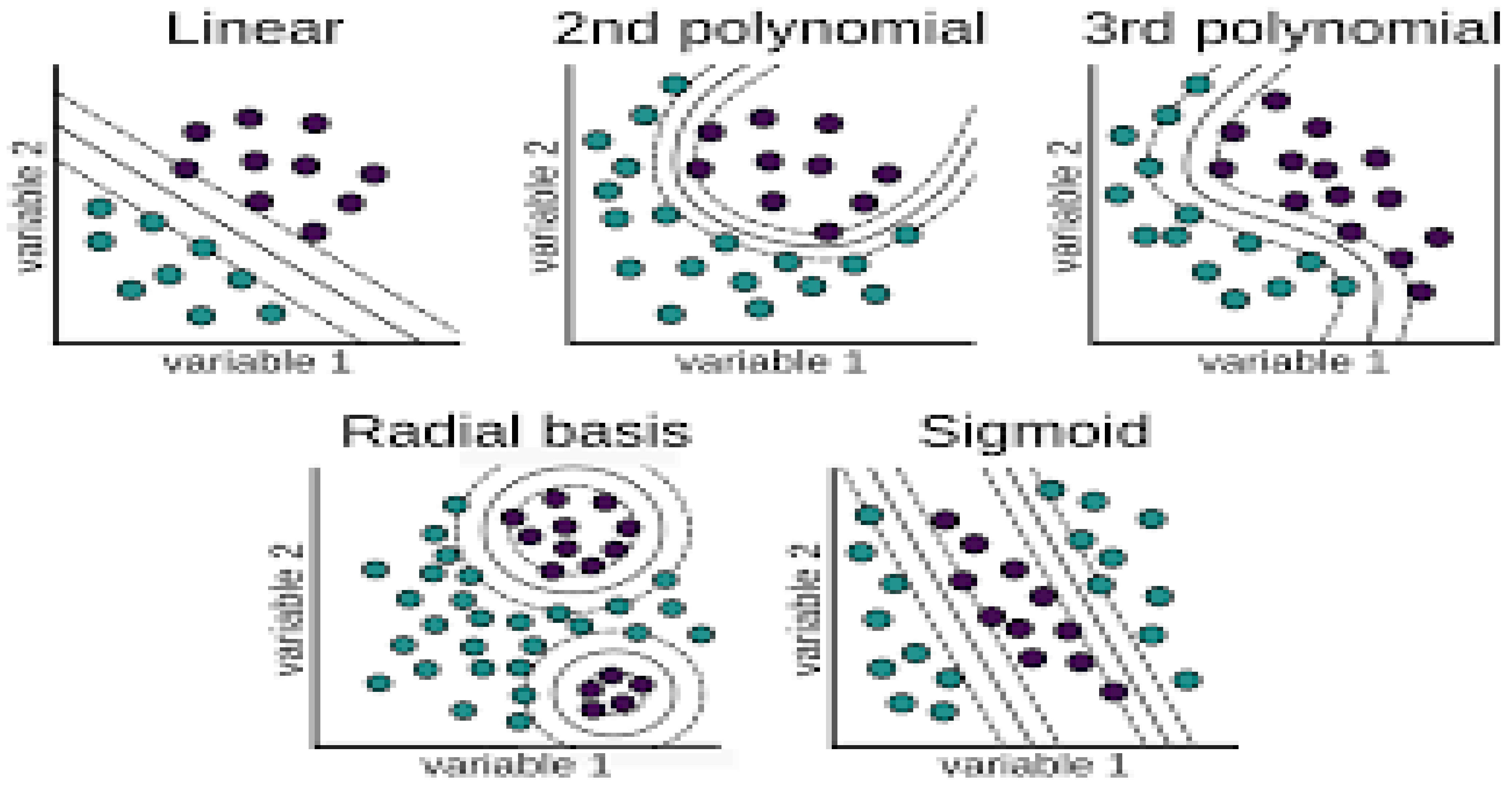
This transformation is called kernel


Input Space

(a)


x1.x2
K(x1.x2)
Φ(x1).Φ(x2)
1.5 1 0.5 0 -0.5 -1 -1.5
X2
-1.5 -1 -0.5 0 0.5 1 1.5
X1
Input Space
3 2 1 0 -1 -2 -3
√2X1X2
0 0.5 1 1.5 2
X1 2
0.5 1 1.5 2 2.5
X2 2
Feature Space

(b)


Linear
2nd polynomial
3rd polynomial
variable 2
variable 1
variable 2
variable 1
variable 2
variable 1
Radial basis
Sigmoid
variable 2
variable 1
variable 2
variable 1

Learning Nonlinear SVM

- Optimization problem:

$$\min_{w} \frac{\|\mathbf{w}\|^2}{2}$$

$$\textit{subject to} \qquad y_i(w \cdot \Phi(x_i) + b) \geq 1, \ \forall\{(x_i, y_i)\}$$

- Which leads to the same set of equations (but involve Φ(x) instead of x)

$$\mathbf{w} = \sum_i \lambda_i y_i \Phi(\mathbf{x}_i)$$

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

$$\lambda_i \{ y_i (\sum_j \lambda_j y_j \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_i) + b) - 1 \} = 0,$$

$$f(\mathbf{z}) = sign(\mathbf{w} \cdot \Phi(\mathbf{z}) + b) = sign(\sum_{i=1}^{n} \lambda_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{z}) + b).$$