# Support Vector Machine

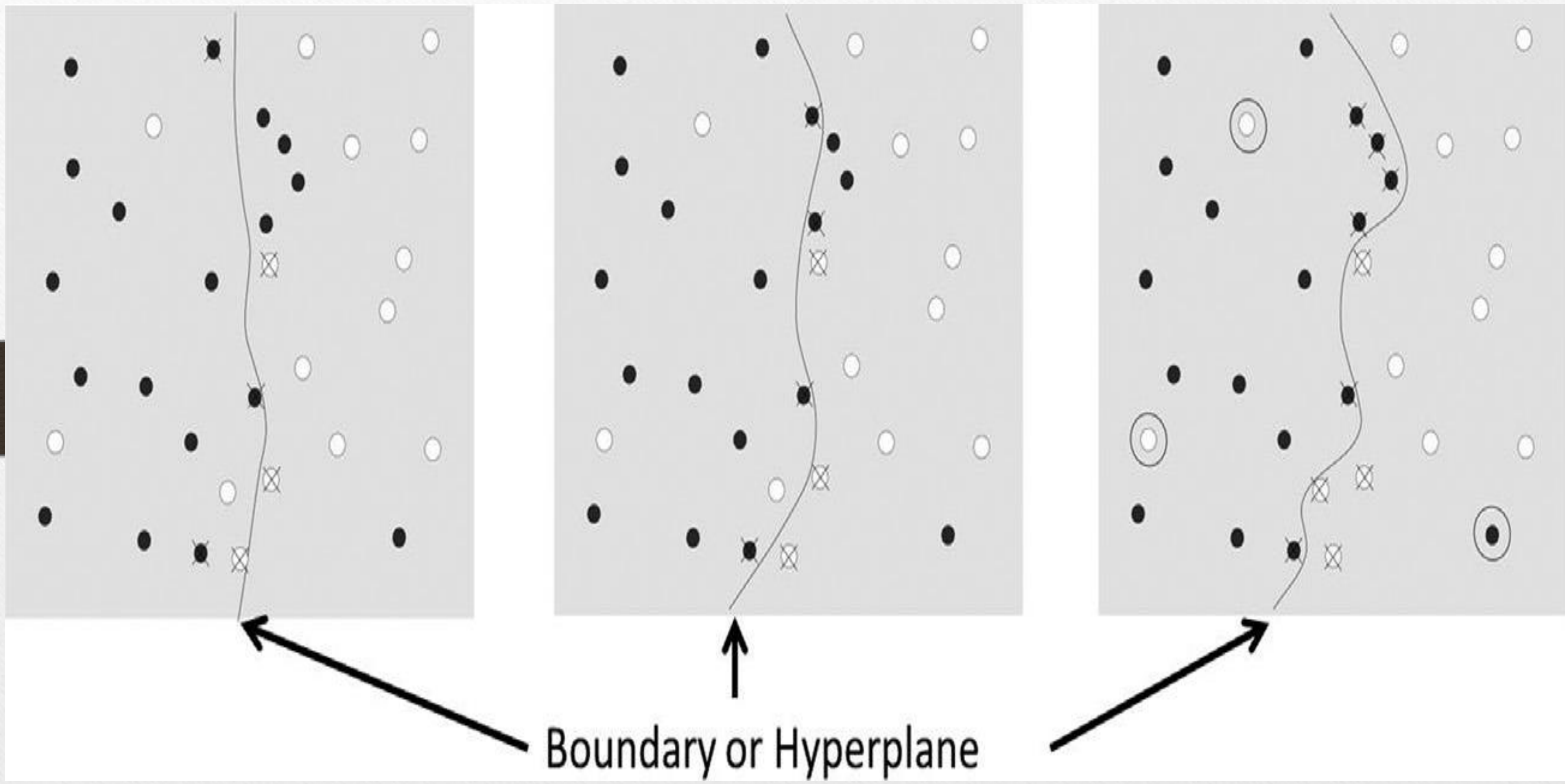
# SUPPORT VECTOR MACHINES

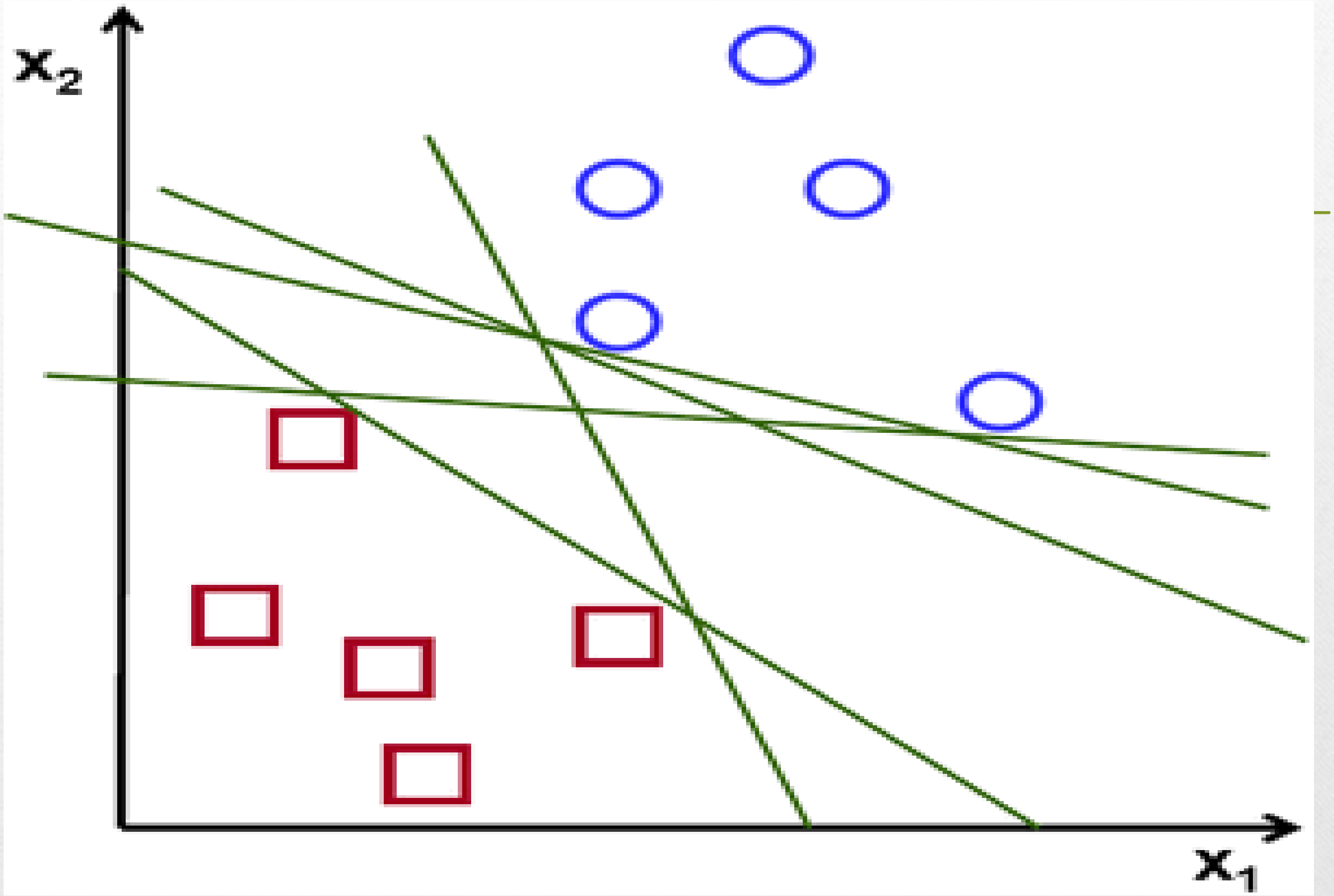
- Most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.
- Primarily, it is used for Classification problems in Machine Learning.
- Support vector machine is highly preferred by many as it produces significant accuracy with less computation power
- There is really no specialized hardware.
- But it is a powerful algorithm that has been quite successful in applications ranging from pattern recognition to text mining.

- SVM emphasizes the interdisciplinary nature of data science by drawing equally from three major areas:
  - computer science,
  - statistics, and
  - Mathematical optimization theory.

- At a basic level, a SVM is a classification method.

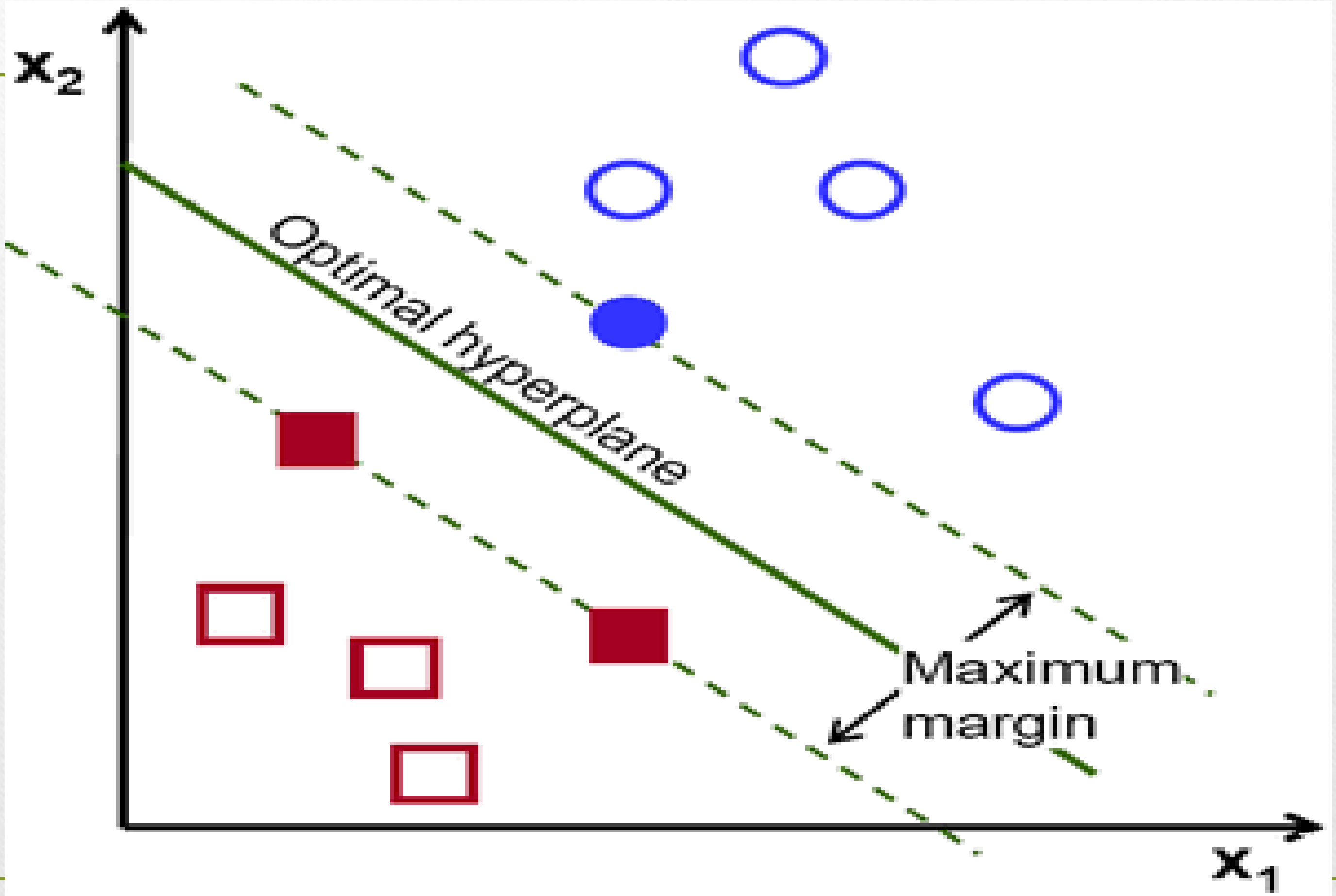
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.
- This best decision boundary is called a hyperplane
- In a simple example of two dimensions, this boundary can be a straight line or a curve.
- The advantage of a SVM is that once a boundary is established, most of the training data is redundant.


Boundary or Hyperplane



x2
x1

# Possible hyperplanes

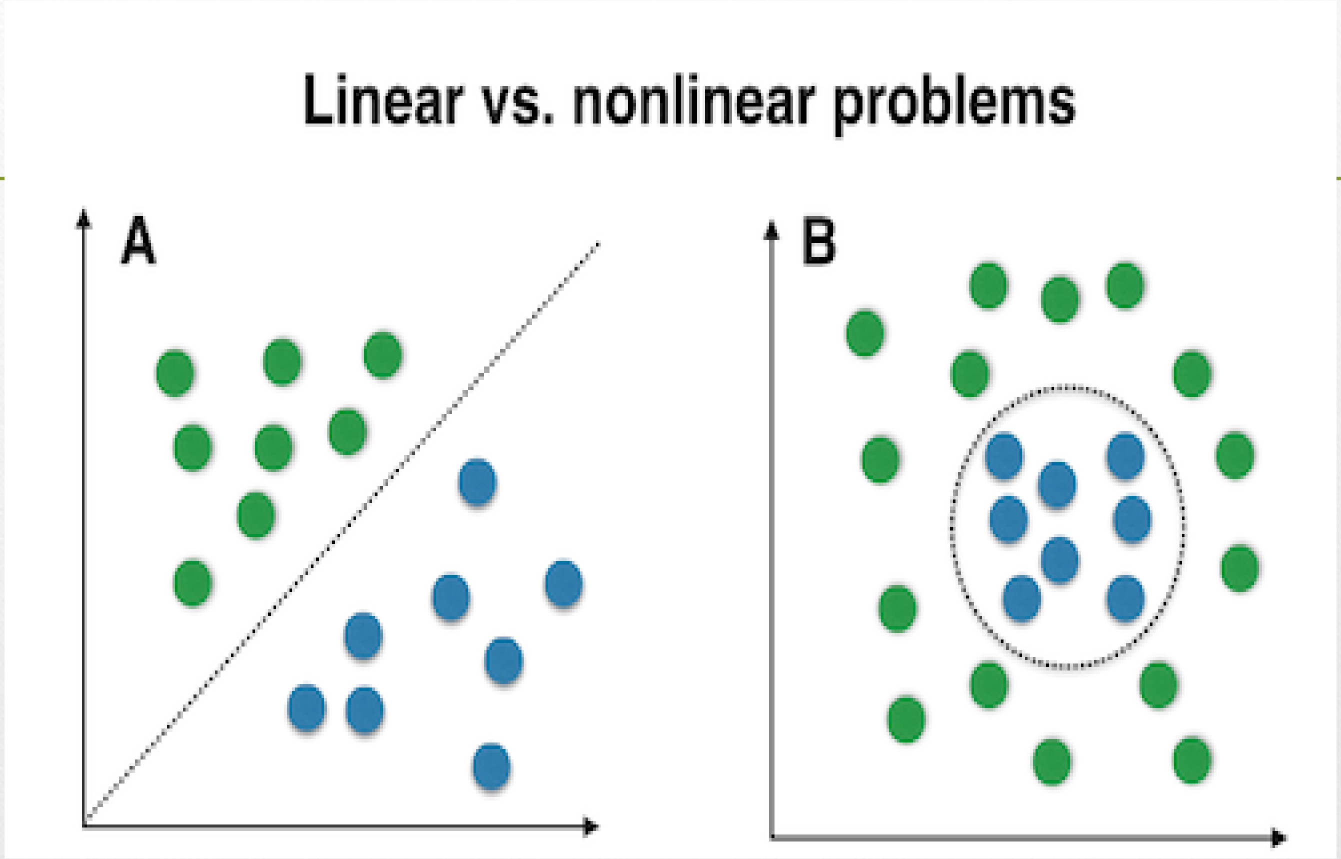


# Important concepts in SVM

- **Support Vectors**
  - Datapoints that are closest to the hyperplane is called support vectors.
  - Separating line will be defined with the help of these data points.
- **Hyperplane**
  - As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.

- **Margin**
  - It may be defined as the gap between two lines on the closet data points of different classes.
  - It can be calculated as the perpendicular distance from the line to the support vectors.
  - Large margin is considered as a good margin and small margin is considered as a bad margin.
  - The **hyperplane** with maximum margin is called the **optimal hyperplane**

- **Linear separability:**
  - A dataset is linearly separable if there is at least one line that clearly distinguishes the classes.
- **Non-linear separability:**
  - A dataset is said to be non-linearly separable if there isn't a single line that clearly distinguishes the classes.


Linear vs. nonlinear problems
A
B

- The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).
- It can be done in the following two steps −
    - First, SVM will generate hyperplanes iteratively that segregates the classes in best way.
    - Then, it will choose the hyperplane that separates the classes correctly.

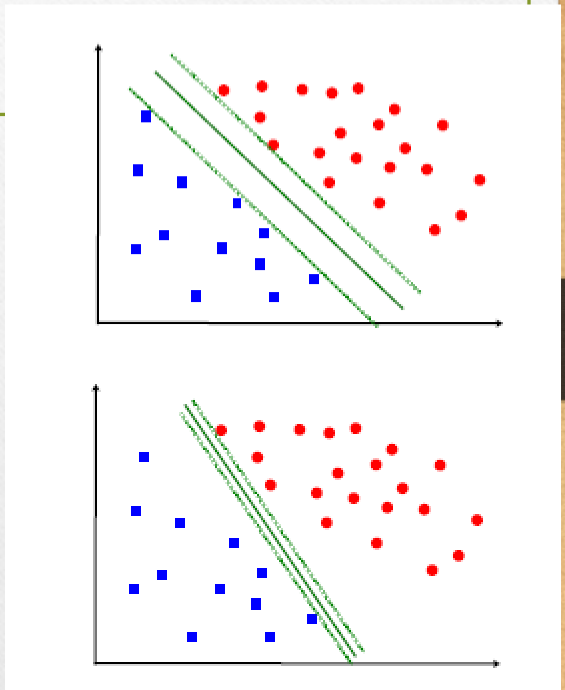
- All it needs is a core set of points that can help identify and fix the boundary.
- These data points are called support vectors because they "support" the boundary.
- A **Support Vector Machine (SVM) can be imagined as a surface that creates** a boundary between points of data plotted in multidimensional that represent examples and their feature values

# What is Support Vector Machine?

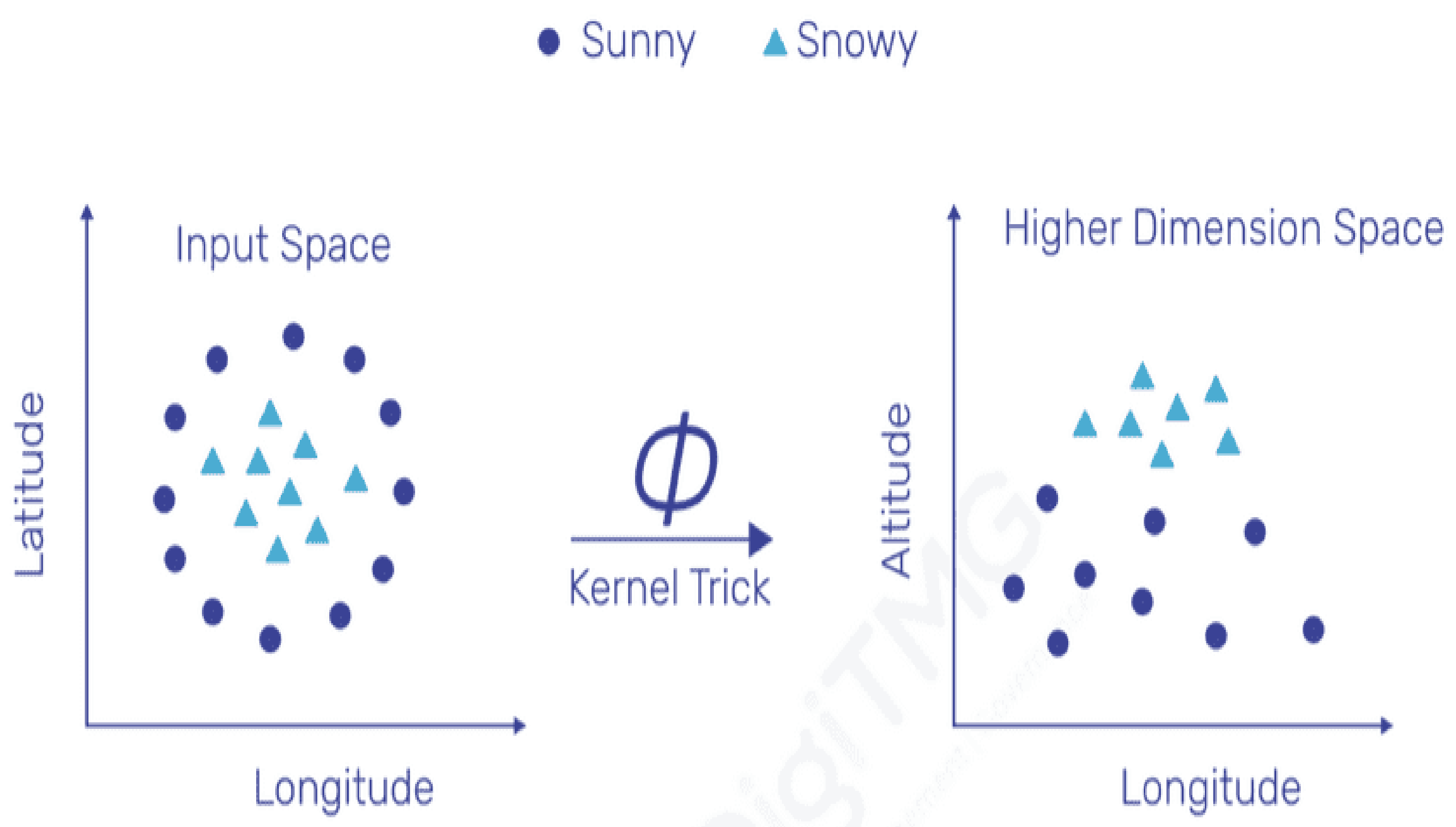
- The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

# Selection of a Good Hyper-Plane

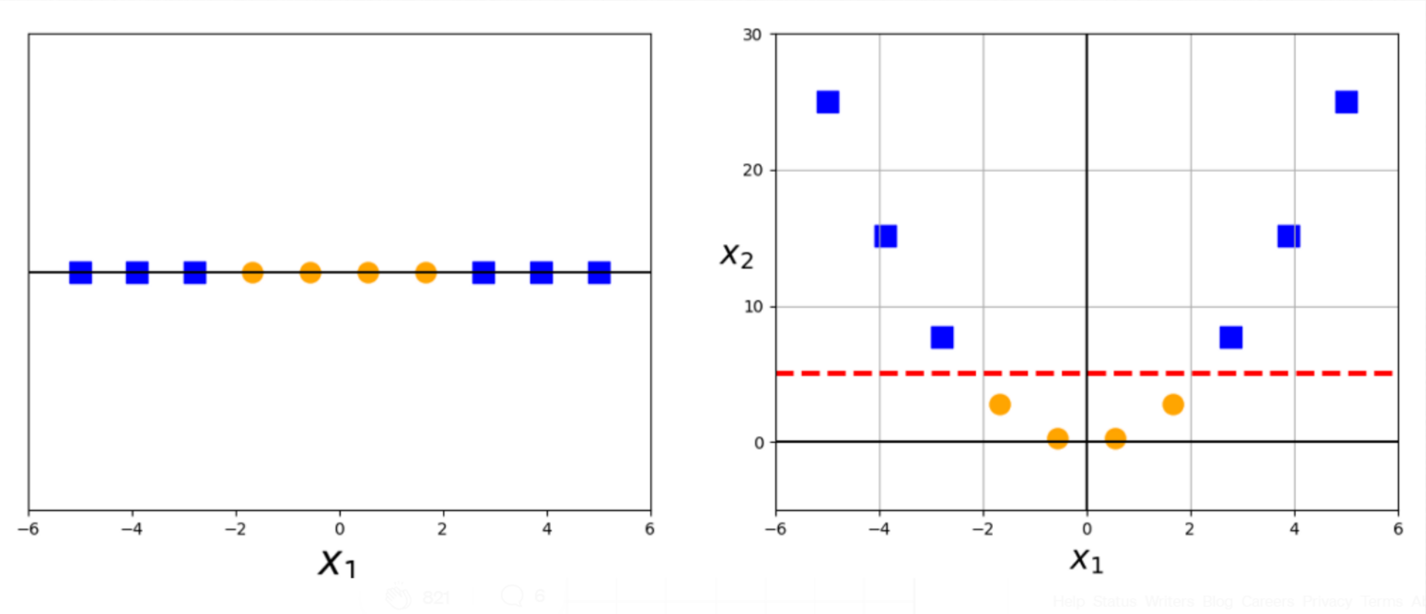
Objective: Select a `good' hyper-plane using only the data!
Intuition: assuming linear separability
(i) Separate the data
(ii) Place hyper-plane `far' from data

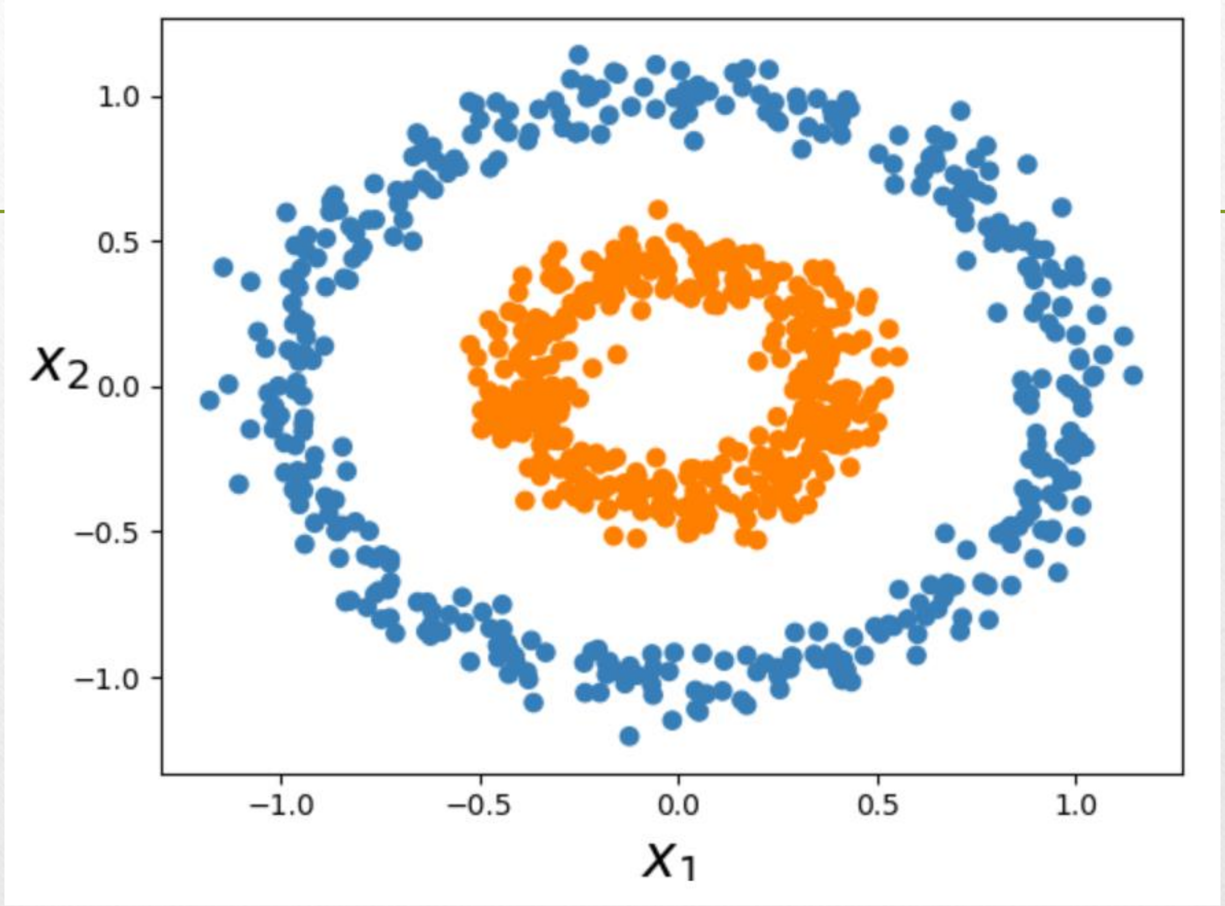


# KERNEL TRICK



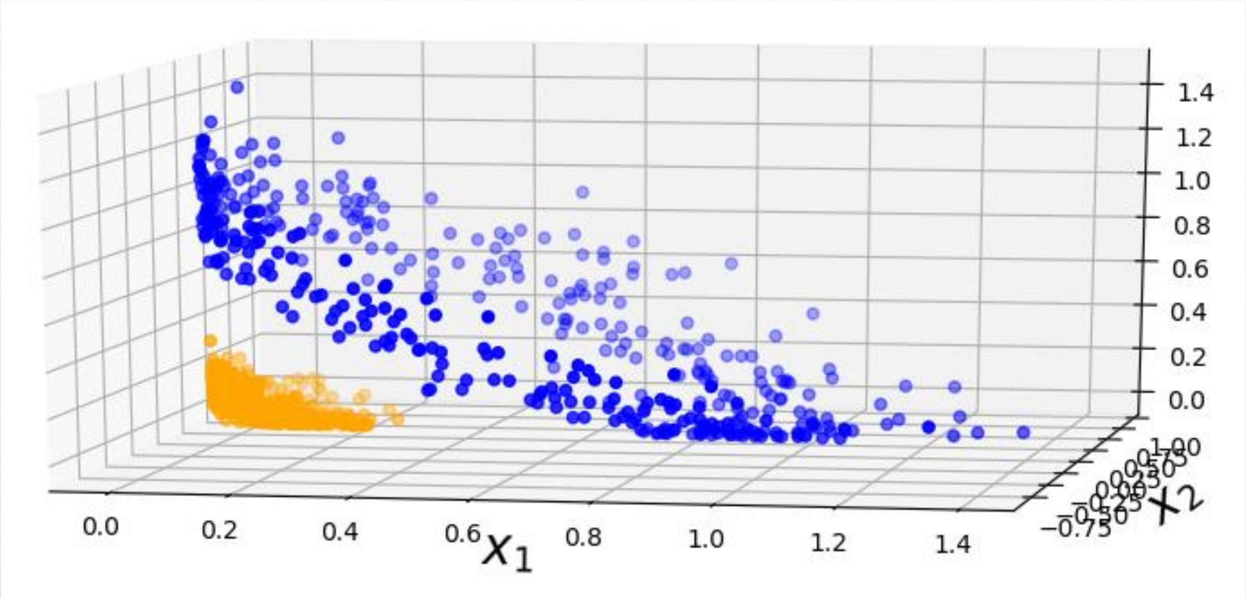
# Kernel Trick




1.0
0.5
$x_2$ 0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
$x_1$

After the transformation, $\phi(\mathbf{x}) = \phi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} {x_1}^2 \\ \sqrt{2}\, x_1 x_2 \\ {x_2}^2 \end{pmatrix}$

The data becomes linearly separable (by a 2-d plane) in 3-dimensions

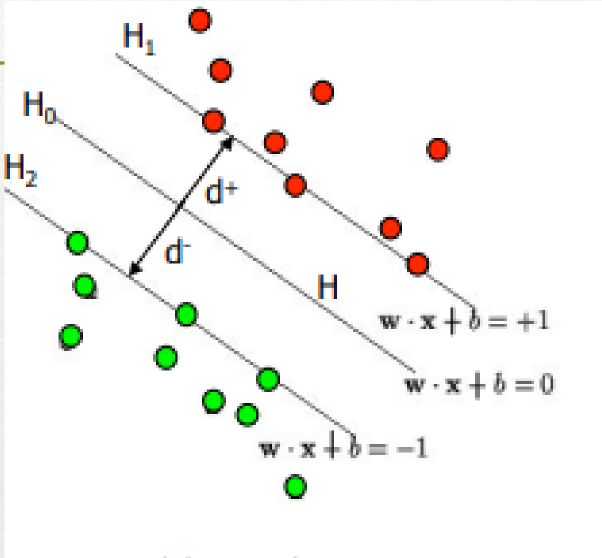


# Definitions

Define the hyperplane H such that:

$\mathbf{x}_i \bullet \mathbf{w}+b = +1$ when $y_i = +1$

$\mathbf{x}_i \bullet \mathbf{w}+b = -1$ when $y_i = -1$




H1 and H2 are the planes:
H1: $\mathbf{x}_i \bullet \mathbf{w}+b = +1$
H2: $\mathbf{x}_i \bullet \mathbf{w}+b = -1$
The points on the planes H1 and H2 are the **Support Vectors**:

$$\{\mathbf{x}_i : |\mathbf{w}^\top \mathbf{x}_i + b| = 1\}$$

d+ = the shortest distance to the closest positive point

d- = the shortest distance to the closest negative point

The margin of a separating hyperplane is $d^+ + d^-$.

# Maximizing the margin

We want a classifier with as big margin as possible.

Recall the distance from a point($x_0$,$y_0$) to a line: $Ax+By+c = 0$ is

$$\mathrm{d} = \frac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$$

The distance between H and H1 is: $\frac{|w.x+b|}{||w||} = \frac{1}{||w||}$

The distance between H1 and H2 is: $\frac{2}{||w||}$

**In order to maximize the margin, we need to minimize ||w||.**
**With the condition that there are no datapoints between H1 and H2:**

$$\left.\begin{array}{l} \mathbf{x}_i \bullet \mathbf{w} + b = +1 \text{ when } y_i = +1 \\ \mathbf{x}_i \bullet \mathbf{w} + b = -1 \text{ when } y_i = -1 \end{array}\right\} \textbf{Can be combined into } \mathbf{y_i(x_i \bullet w) = 1}$$

# Primal Form of Optimization

- Constrained Optimization Problem
- Maximize $\frac{2}{\|\theta\|}$ where $\|\theta\| = \sqrt{\theta_1^2 + \theta_2^2 + \cdots + \theta_d^2}$

- Thus

  Minimize $\frac{1}{2}\|\theta\|$

i.e.

Minimize $\frac{1}{2}\|\theta\|^2$

- Subject to constraints

  $y_i(\theta^T x_i) \geq 1, \forall i$

Primal Version

Quadratic Programming Problem

# Lagrangian Function

Let $f(x)$ be a function to be optimized subject to inequality constraints

$$Minimize\ f(x)$$

Subject to $g(x) \geq 0$

Convert inequality constraints to equality constraints

$$g(x) - s^2 = 0$$

$$L(x, \alpha) = f(x) - \alpha(g(x) - s^2)$$

In SVM Context

$f(x) = \frac{1}{2}||\theta||^2$ subject to constraints $g(x) - s^2 = y_i(\theta^T x_i) - 1 = 0, \forall i$

# Optimization using Lagrangian Multipliers

$$L(\theta, \alpha) = \frac{1}{2}\Sigma_{j=1}^{d}\theta_j^2 \; - \Sigma_{i=1}^{m}\alpha_i(y_i\theta^T X_i - 1)$$

OR

$$L(\theta_0, \theta, \alpha) = \frac{1}{2}\Sigma_{j=1}^{d}\theta_j^2 \; - \Sigma_{i=1}^{m}\alpha_i(y_i(\theta^T X_i + \theta_0) - 1)$$



Such that $\alpha_i \geq 0, \forall i$

Minimize over θ and Maximize over α.

Primal Lagrangian : $\max\limits_{\alpha} \min\limits_{\theta_0,\theta} L(\theta_0, \theta, \alpha)$

# Solve Lagrangian Function

$$L(\theta_0, \theta, \alpha) = \frac{1}{2}\Sigma_{j=1}^{d}\theta_j^2 \; - \Sigma_{i=1}^{m}\alpha_i(y_i(\theta^T X_i + \theta_0) - 1) \quad \text{(Eqn 1)}$$

Such that $\alpha_i \geq 0, \forall i$

To solve, set $\frac{\partial L}{\partial \theta_0} = 0, \frac{\partial L}{\partial \theta} = 0, \frac{\partial L}{\partial \alpha} = 0$

$$\frac{\partial L}{\partial \theta_0} = 0 \rightarrow -\Sigma_{i=1}^{m}\alpha_i y_i = 0 \text{ (Eqn 2)}$$

$$\frac{\partial L}{\partial \theta} = 0 \rightarrow \theta = \Sigma_{i=1}^{m}\alpha_i y_i X_i \text{ (Eqn 3)}$$

# Solve Lagrangian Function

$$L(\theta_0, \theta, \alpha) = \frac{1}{2}\Sigma_{j=1}^{d}\theta_j^2 \; - \Sigma_{i=1}^{m}\alpha_i(y_i(\theta^T X_i + \theta_0) - 1) \text{ (Eqn.1)}$$

Such that $\alpha_i \geq 0, \forall i$

Expanding….

$$\frac{1}{2}\theta^2 \; - \Sigma_{i=1}^{m}\alpha_i(y_i\theta^T X_i + y_i\theta_0) - \alpha_i)$$

$$= \frac{1}{2}\theta^2 \; - \Sigma_{i=1}^{m}\alpha_i . y_i\theta^T X_i + \alpha_i y_i\theta_0 + \Sigma_{i=1}^{m}\alpha_i \text{ (Eqn 4)}$$

# Towards Dual Form

Substitute

$$-\Sigma_{i=1}^{m}\alpha_i y_i = 0 \text{ (Eqn 2)}$$

$$\theta = \Sigma_{i=1}^{m}\alpha_i y_i X_i \text{ (Eqn 3)}$$

*Into*

$$\frac{1}{2}\theta^2 - \Sigma_{i=1}^{m}\alpha_i . y_i\theta^T X_i - \Sigma_{i=1}^{m}\alpha_i y_i \theta_0 + \Sigma_{i=1}^{m}\alpha_i \text{ (Eqn 4)}$$

We get

$$\frac{1}{2}\theta^2 - \theta^2 + \Sigma_{i=1}^{m}\alpha_i = -\frac{1}{2}\theta^2 + \Sigma_{i=1}^{m}\alpha_i$$

New Objective Function

$$J(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \langle X_i, X_j\rangle$$

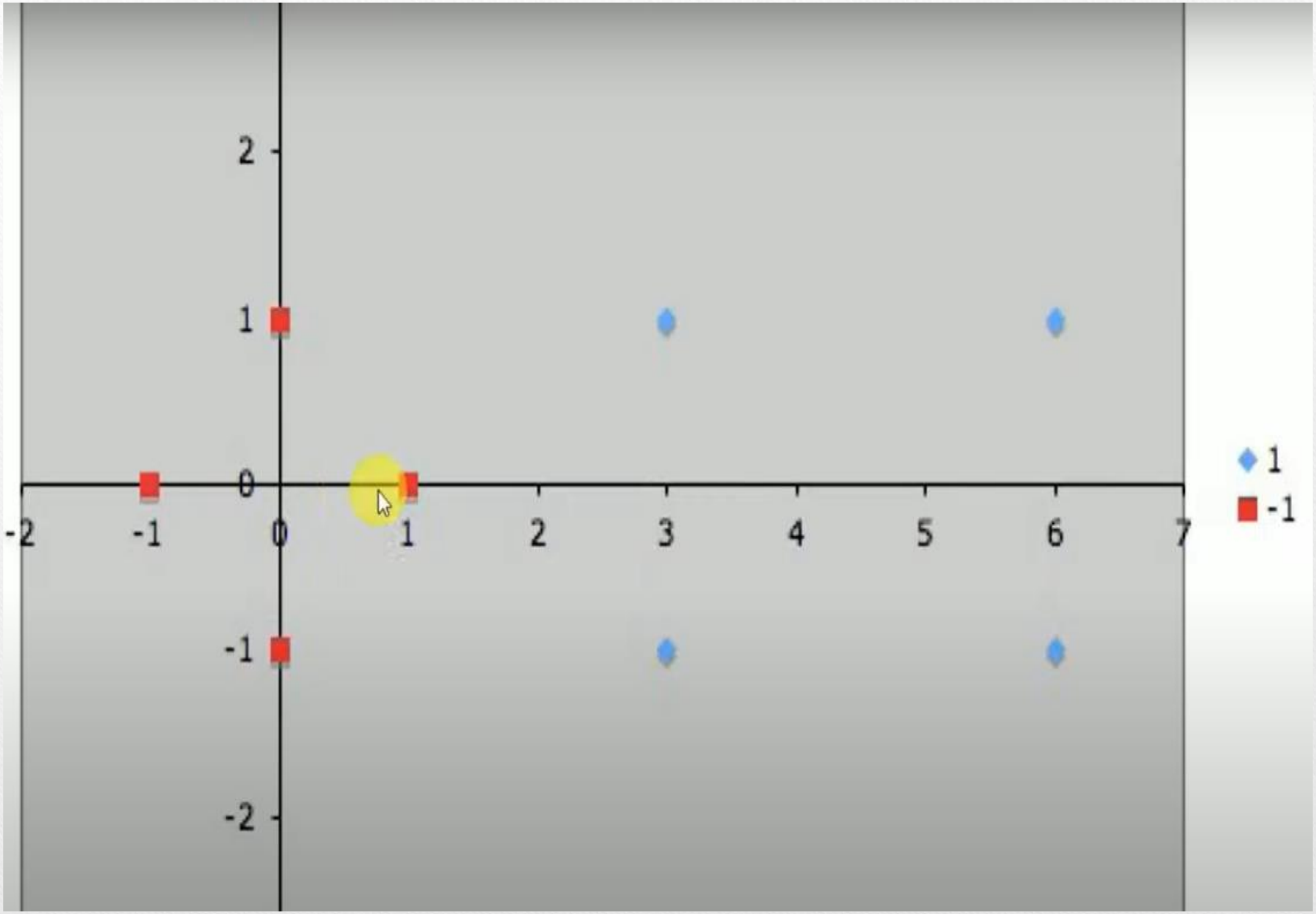
# Support Vector Machine - Linear Example Solved

Suppose we are given the following positively labeled data points,

$$\left\{ \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix} \right\}$$

and the following negatively labeled data points,

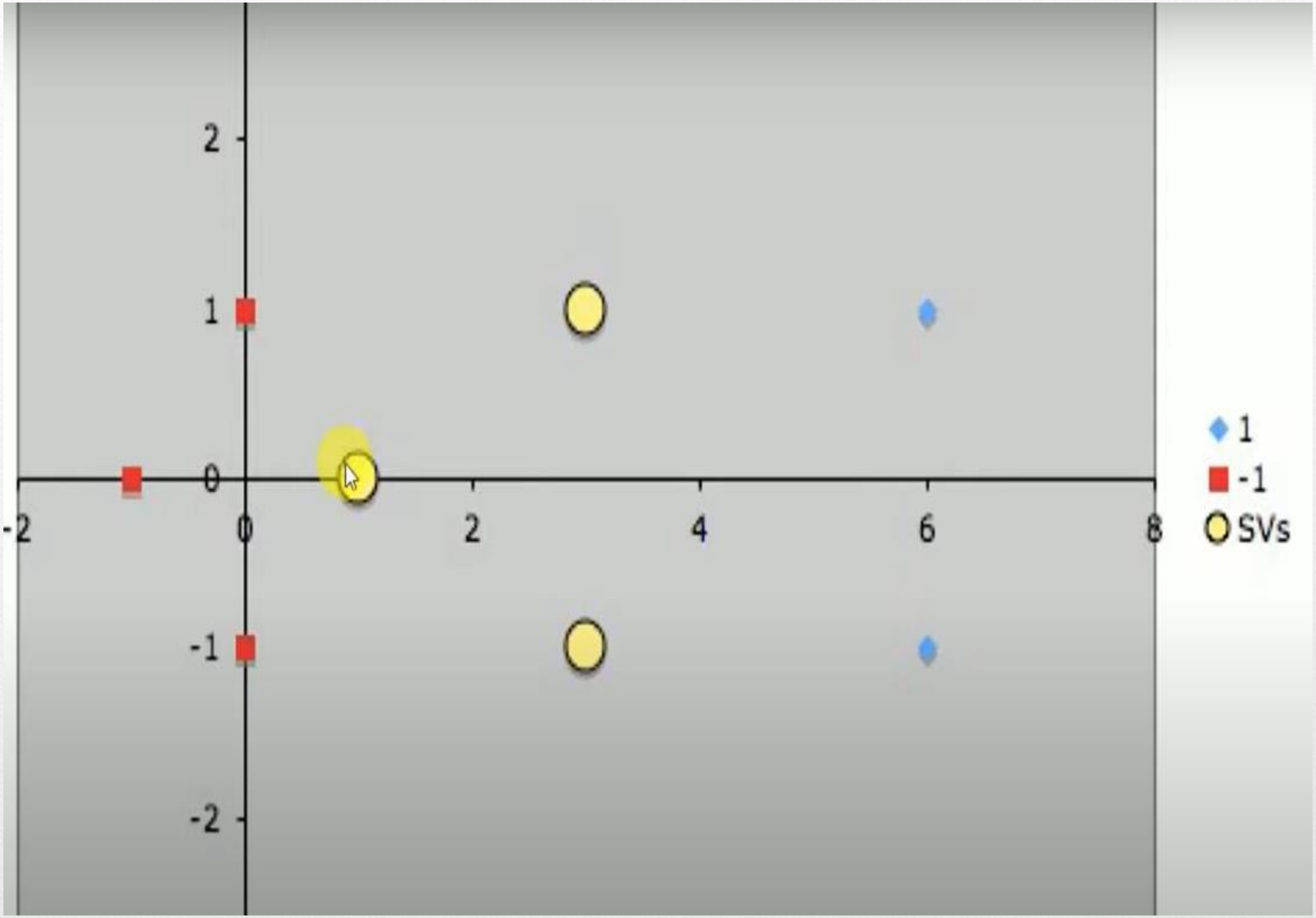
$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix} \right\}$$


2
1
0
-1
-2
-2
-1
0
1
2
3
4
5
6
7
1
-1

# Support Vector Machine - Linear Example Solved

- By inspection, it should be obvious that there are **three** support vectors,

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix} \right\}$$



2
1
0
-1
-2
-2
0
2
4
6
8
1
-1
SVs

# Support Vector Machine - Linear Example Solved

- Each vector is augmented with a 1 as a bias input

- So, $s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, then $\tilde{s_1} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$

- Similarly,

- $s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}$, then $\tilde{s_2} = \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix}$ *and* $s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$, then $\tilde{s_3} = \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix}$

# Support Vector Machine - Linear Example Solved

$$
\begin{aligned}
\alpha_1 \tilde{s_1} \cdot \tilde{s_1} + \alpha_2 \tilde{s_2} \cdot \tilde{s_1} + \alpha_3 \tilde{s_3} \cdot \tilde{s_1} &= -1 \\
\alpha_1 \tilde{s_1} \cdot \tilde{s_2} + \alpha_2 \tilde{s_2} \cdot \tilde{s_2} + \alpha_3 \tilde{s_3} \cdot \tilde{s_2} &= +1 \\
\alpha_1 \tilde{s_1} \cdot \tilde{s_3} + \alpha_2 \tilde{s_2} \cdot \tilde{s_3} + \alpha_3 \tilde{s_3} \cdot \tilde{s_3} &= +1
\end{aligned}
$$

$$\alpha_1\begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix}+\alpha_2\begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix}+\alpha_3\begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix}=-1$$

$$\alpha_1\begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix}+\alpha_2\begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix}+\alpha_3\begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\1\\1\end{pmatrix}=1$$

$$\alpha_1\begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix}+\alpha_2\begin{pmatrix}3\\1\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix}+\alpha_3\begin{pmatrix}3\\-1\\1\end{pmatrix}\begin{pmatrix}3\\-1\\1\end{pmatrix}=1$$

$$\alpha_1(1+0+1)+\alpha_2(3+0+1)+\alpha_3(3+0+1)=-1$$

$$\alpha_1(3+0+1)+\alpha_2(9+1+1)+\alpha_3(9-1+1)=1$$

$$\alpha_1(3+0+1)+\alpha_2(9-1+1)+\alpha_3(9+1+1)=1$$

$$2\alpha_1 + 4\alpha_2 + 4\alpha_3 = -1$$

$$4\alpha_1 + 11\alpha_2 + 9\alpha_3 = 1$$

$$4\alpha_1 + 9\alpha_2 + 11\alpha_3 = 1$$

$$\alpha_1 = -3.5$$

$$\alpha_2 = 0.75$$

$$\alpha_3 = 0.75$$

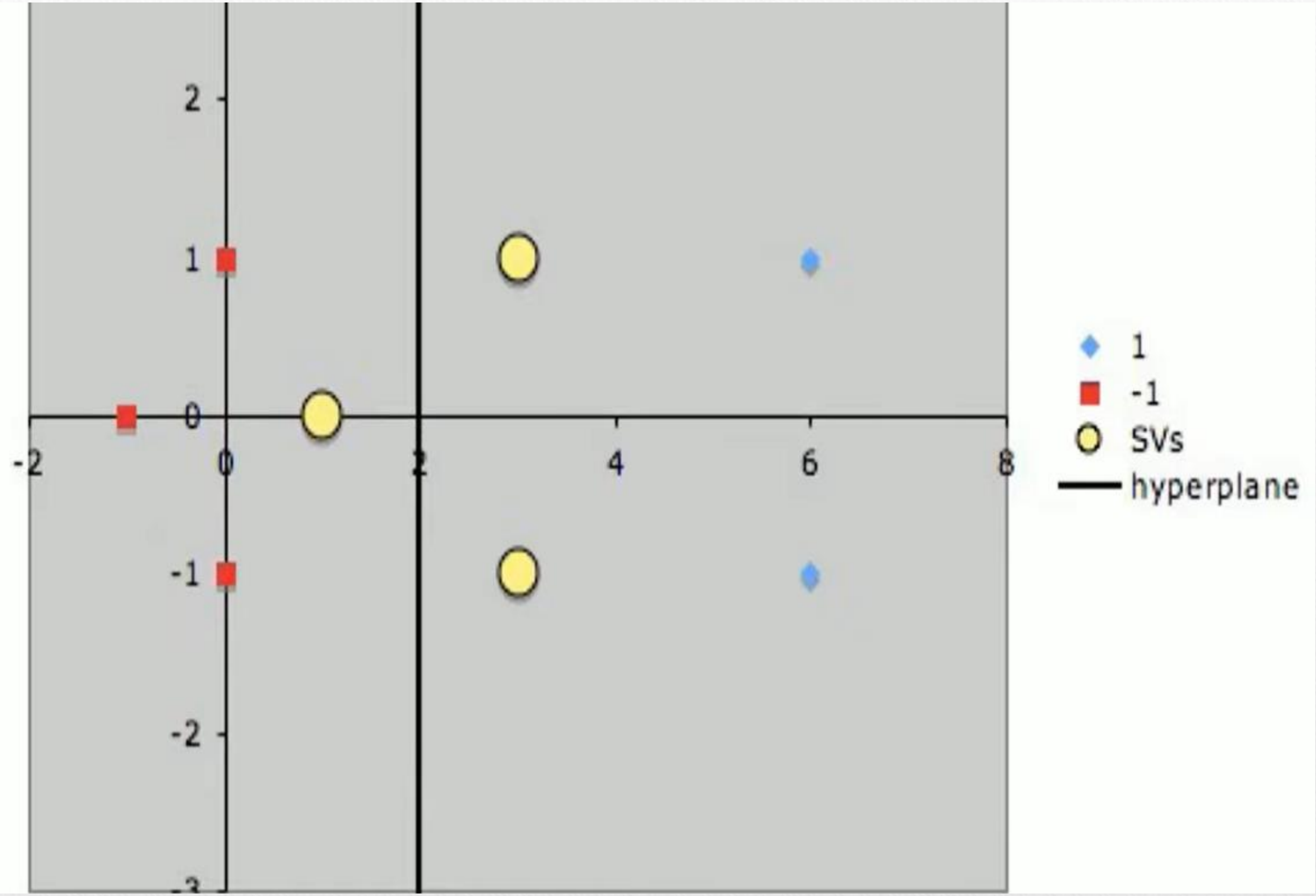
# Support Vector Machine - Linear Example Solved

$$\tilde{w} = \sum_i \alpha_i \tilde{s}_i$$

$$= -3.5 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix}$$

- Finally, remembering that our vectors are augmented with a bias.
- We can equate the last entry in $\widetilde{w}$ as the hyperplane offset b and write the separating
- Hyperplane equation $\boldsymbol{y} = \boldsymbol{wx} + \boldsymbol{b}$
- with $\boldsymbol{w} = \binom{1}{0}$ and $\boldsymbol{b} = -2$.


2
1
0
-1
-2
-2
0
2
4
6
8
1
-1
SVs
hyperplane

- *Positively Labeled data points*
  - $\begin{pmatrix}1\\1\end{pmatrix}\begin{pmatrix}1\\-1\end{pmatrix}\begin{pmatrix}2\\1\end{pmatrix}\begin{pmatrix}2\\-1\end{pmatrix}$
- Negatevely Labeled data points
  - $\begin{pmatrix}4\\0\end{pmatrix}\begin{pmatrix}5\\1\end{pmatrix}\begin{pmatrix}5\\-1\end{pmatrix}\begin{pmatrix}6\\0\end{pmatrix}$

# Popular SVM Kernel functions:

1. Linear Kernel: It is just the dot product of all the features. It doesn't transform the data.
2. Polynomial Kernel: It is a simple non-linear transformation of data with a polynomial degree added.
3. Gaussian Kernel: It is the most used SVM Kernel for usually used for non-linear data.
4. Sigmoid Kernel: It is similar to the Neural Network with sigmoid activation function.

- Common kernel functions for SVM
  - linear $\qquad k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \cdot \mathbf{x}_2$
  - polynomial $\qquad k(\mathbf{x}_1, \mathbf{x}_2) = (\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)^d$
  - Gaussian or radial basis $\qquad k(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma \left\| \mathbf{x}_1 - \mathbf{x}_2 \right\|^2\right)$
  - sigmoid $\qquad k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma\, \mathbf{x}_1 \cdot \mathbf{x}_2 + c)$

# Pros and Cons associated with SVM

- **Pros:**
  - It works really well with a clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where the number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- **Cons:**
  - It doesn't perform well when we have large data set because the required training time is higher
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
  - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.

# Problems that can be solved using SVM

- Image classification
- Recognizing handwriting
- Caner detection

# Difference between SVM and Logistic Regression

- SVM tries to finds the “best” margin (distance between the line and the support vectors) that separates the classes and this reduces the risk of error on the data, while logistic regression does not, instead it can have different decision boundaries with different weights that are near the optimal point.

- SVM works well with unstructured and semi-structured data like text and images while logistic regression works with already identified independent variables.
- SVM is based on geometrical properties of the data while logistic regression is based on statistical approaches.
- The risk of overfitting is less in SVM, while Logistic regression is vulnerable to overfitting.

# When To Use Logistic Regression vs Support Vector Machine

- Depending on the number of training sets (data)/features that you have, you can choose to use either logistic regression or support vector machine.
    - Lets take these as an example where :
      $n$ = *number of features,*
      $m$ = *number of training examples*

- 1. If *n is large (1–10,000) and m is small (10–1000)* : use logistic regression or SVM with a linear kernel.
- 2. If *n is small (1–10 00) and m is intermediate (10–10,000*) : use SVM with (Gaussian, polynomial etc) kernel
- 3. If *n is small (1–10 00), m is large* (50,000–1,000,000+): first, manually add more features and then use logistic regression or SVM with a linear kernel