Data Science and Artificial Intelligence

Machine Learning

Support Vector Machine

Lecture No. 3

By- SIDDHARTH SABHARWAL SIR

Recap of Previous Lecture

Topic — svm

Topic — algo

Topic — lagrangian

Topic

Topic

Turn on Slide map

Topics to be Covered

Topic
Topic
Topic
Topic
Topic

SVM

Paimal / dual / Kernels.


NEVER STOP
BELIEVING
IN HOPE
BECAUSE
MIRACLES
HAPPEN
EVERY DAY
blurmark.com

REVISION

SVM (algorithm)

↓

Scaling

$wx+b=0$

$wx+b=\pm 1$.

algo $\min \frac{1}{2}\|w\|^2$

s.t. $y_i(w x_i+b) \geq 1$

REVISION

SVM (algorithm)

Lagrangian

$$\min f(x) \quad \text{st. } g(x) \leq 0$$

$$\rightarrow \min_{x} \left(\max_{\lambda} f(x) + \lambda g(x) \right) \quad \text{st. } \lambda \geq 0$$

SVM (algorithm)

dagnanqian

- $\min f(x)$ $\Rightarrow$ $\min_{x} \left(\max_{\lambda_1, \lambda_2, \lambda_3 \dots} f(x) + \sum_{i=1}^{N} \lambda_i g_i(x) \right)$

$g_1(x) \leq 0$

$g_2(x) \leq 0$

$\vdots$

$\Rightarrow$ N: Number of Conditions

$\Rightarrow$ N: λ's

Steps in Support Vector Machine

Lets see the lagrangian...

So the algo

$\min \frac{1}{2}\|\omega\|^2$

St. $y_i(\omega x_i + b) \geq 1$

St. $\Rightarrow$ $\boxed{1 - y_i(\omega x_i + b) \leq 0}$

- So we have N number of points

So we will have N number of conditions

Basic algo.

SVMs

Steps in Support Vector Machine

Lets see the lagrangian...

$$\Rightarrow \underbrace{\left\{ \min_{(\omega)} \left[\max_{\lambda_i} \left(\frac{1}{2}\|\omega\|^2 + \sum_{i=1}^{N} \lambda_i \left(1-(y_i(\omega x_i + b))\right) \right) \right] \quad \text{St. } \lambda_i \geq 0 \right.}$$

$\Rightarrow$ Primal formulation of svm

SVMs

No need to Remember ✗

Support Vector Machine

SVM primal equation

Primal and dual formulation

Problem $\left(\begin{array}{l} \min f(x) \\ \text{St. } g(x) \leq 0 \end{array} \right) \Rightarrow$ Primal $\min\limits_{x} \left(\max\limits_{\lambda} \underbrace{f(x) + \lambda g(x)}_{L} \right)$

St. $\lambda \geq 0$

$\rightarrow$ Dual $\max\limits_{\lambda} \left(\min\limits_{x} f(x) + \lambda g(x) \right)$

(Karush Kuhn Tucker) X (Rote learning)

Support Vector Machine

SVM primal equation

⇒ KKT conditions (if x^* is the solution then)

1) $\left.\frac{\partial L}{\partial x}\right|_{x=x^*} = 0$

2) $g(x^*) \leq 0$

3) $\lambda g(x^*) = 0$

(Remember X)

Support Vector Machine

SVM primal equation

So Problem $\Rightarrow$

$$\min_{\omega} \max_{\lambda_i} \Big\{ \overbrace{\underbrace{\frac{1}{2}\|\omega\|^2 + \sum_{i=1}^{N} \lambda_i \big(1 - y_i(\omega x_i + b)\big)}_{\text{we have to optimize this}}}^{\mathcal{L}} \Big\}$$

ω^* is solution

b^* is solution

$$\frac{\partial L}{\partial \omega} = 0 \Rightarrow \omega + \sum_{i=1}^{N} \lambda_i (-y_i x_i) = 0$$

$$\Rightarrow \omega = \sum_{i=1}^{N} \lambda_i y_i x_i$$

SVMs

Support Vector Machine

SVM primal equation

So Problem ⇒

$$\min_{\omega} \max_{\lambda_i} \underbrace{\left\{ \overbrace{\frac{1}{2}\|\omega\|^2 + \sum_{i=1}^{N} \lambda_i \left(1 - y_i(\omega x_i + b)\right)}^{\mathcal{L}} \right]}_{\text{we have to optimize this}}$$

ω^* is solution

b^* is solution

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \lambda_i y_i = 0$$

SVMs

Remember x

Steps in Support Vector Machine

KKT

1) $\frac{\partial L}{\partial \omega} = 0$, $\omega = \sum_{i=1}^{N} \lambda_i y_i x_i$

2) $\frac{\partial L}{\partial b} = 0$, $\sum_{i=1}^{N} \lambda_i y_i = 0$

3) $1 - y_i(\omega x_i + b) \leq 0$

4) $\lambda_p (1 - y_i(\omega x_i + b)) = 0$

Lets see the lagrangian...

$\Rightarrow A.B = 0$

- $A = 0, B = 0$ ✓
- if $A \neq 0$ then $B = 0$
- if $B \neq 0$ then $A = 0$

The Lagrangian

The primal formulation

Imp

$\lambda_i \left(1 - y_i(\omega x_i + b)\right) = 0$

- for Support vectors $\omega x_i + b = \pm 1$

  $y_i(\omega x_i + b) = 1$

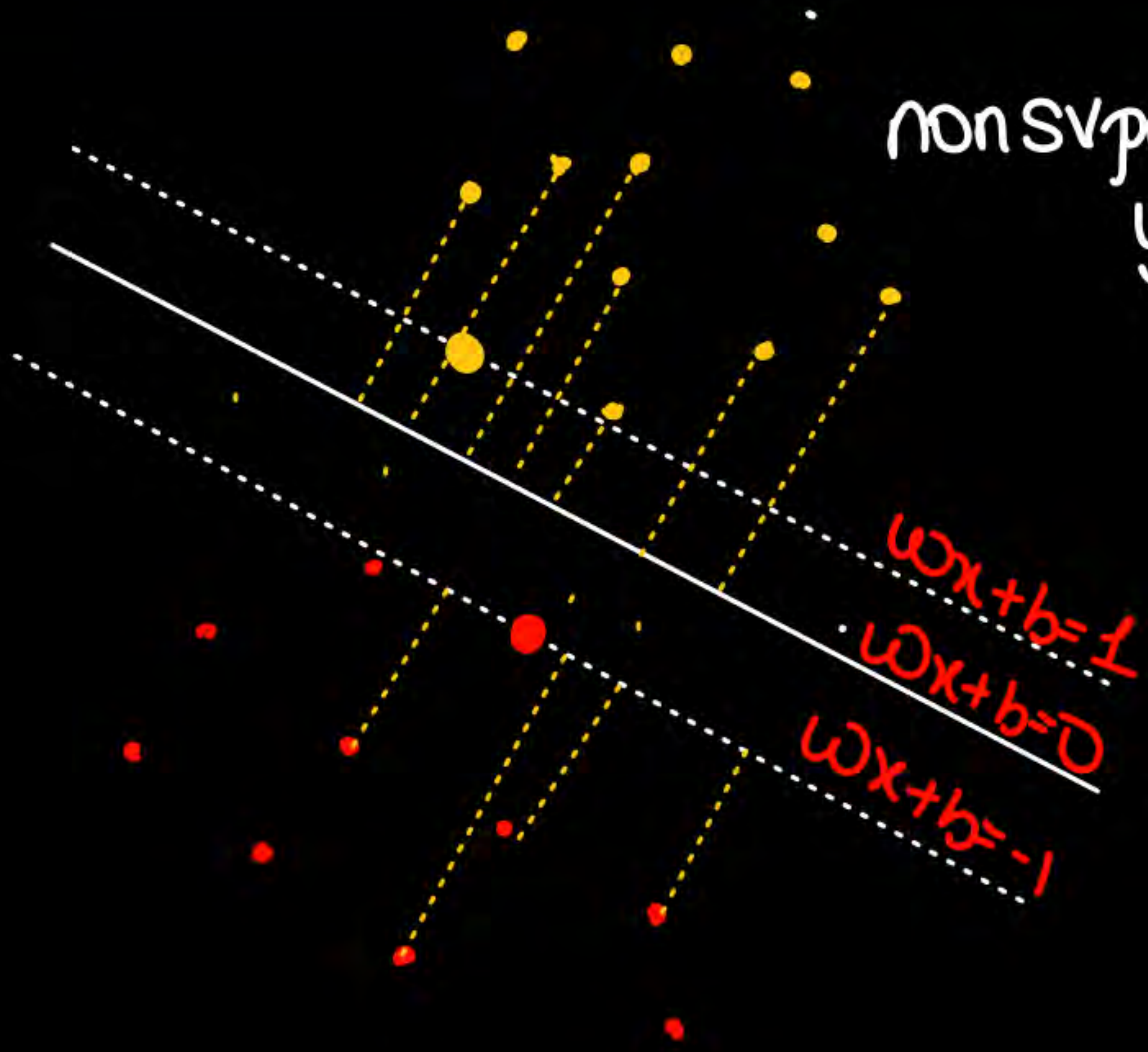
  So for sv $1 - y_i(\omega x_i + b) = 0$, So λ_i can be non zero for sv

- If any point is not SV $y_i(\omega x_i + b) > 1$

  then $1 - y_i(\omega x_i + b) \neq 0$ So $\lambda_i = 0$ for non sv's.

The Lagrangian

The primal formulation



non sv points

$y_i(\omega x_i+b)>1$

So $\lambda_i=0$ for nonsv

$\lambda_i \neq 0$ for sv

$\Rightarrow \omega=\sum_{i=1}^{N}\lambda_i y_i x_i$

$\Rightarrow$ So in svm the classifier is decided by sv only.

SVMs

(norm w) $\|w\| = \sqrt{w_1^2 + w_2^2 + w_3^2 + \cdots w_D^2}$

The Lagrangian

What is the dual formulation...

- w = is not a single parameter
  if we have D number of dimension
  then $w = [w_1, w_2\, w_3 \cdots w_D]$
- $w_1^2 + w_2^2 + w_3^2 + \cdots w_D^2 = \|w\|^2 \Rightarrow (w\, w^T)$

$$[w_1\, w_2\, w_3 \cdots w_D] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

$(w\ w^T)$

$\Rightarrow (w_1^2 + w_2^2 + w_3^2 \cdots)$

SVMs

The Lagrangian

What is the dual formulation...

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_D \end{bmatrix}$$

then $w^T w$

$$\Rightarrow \begin{bmatrix} w_1 & w_2 & \cdots & w_D \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix}$$

$$\Rightarrow w_1^2 + w_2^2 + \cdots w_D^2 \Rightarrow \|w\|^2$$

The Lagrangian

What is the dual formulation...

Now

Primal formulation

$$\Rightarrow \min_{\omega} \max_{\lambda_i} \left(\frac{1}{2}\|\omega\|^2 + \sum_{i=1}^{N} \lambda_i \left(1 - y_i(\omega x_i + b)\right) \right)$$

$\|\omega\|^2 \rightarrow \omega\omega^T$

$$\Rightarrow \omega = \sum_{i=1}^{N} \lambda_i y_i x_i$$

each x_i has D dimensions

$$\omega = [\omega_1 \; \omega_2 \; \omega_3 \; \cdots \; \omega_D]$$

$$x_i = [x_i^1, x_i^2, \cdots x_i^D]$$

The Lagrangian

So $wx_i + b \Rightarrow (w_1 x_i^1 + w_2 x_i^2 + w_3 x_i^3 - - - w_D x_i^D) + b$

$\cancel{(w^T x_i + b)}$

$(w x_i^T + b)$

The solution of primal is also the solution of the dual

The Lagrangian

So

$$\min_{\omega} \max_{\lambda} \left[\frac{1}{2}\omega\omega^T + \sum_{i=1}^{N} \lambda_i \left(1 - y_i(\omega x_i^T + b)\right) \right]$$

$$\omega = \sum_{j=1}^{N} \lambda_j y_j x_j$$

$$\max_{\lambda} \left[\frac{1}{2} \sum_{j=1}^{N} \lambda_j y_j x_j \left(\sum_{i=1}^{N} \lambda_i y_i x_i \right)^T + \sum_{i=1}^{N} \lambda_i \left(1 - y_i \sum_{j=1}^{N} \lambda_j y_j x_j \cdot x_i^T - y_i b\right) \right.$$

$$\max_{\lambda} \left[\frac{1}{2} \sum_{j=1}^{N} \lambda_j y_j x_j \sum_{i=1}^{N} \lambda_i y_i x_i^T + \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{N} \lambda_i y_i x_i^T \sum_{j=1}^{N} \lambda_j y_j x_j - b \cancel{\sum_{i=1}^{N} \lambda_i y_i} \right]$$

SVMs

The Lagrangian

So $\max_{\lambda} \left[\sum_{i=1}^{N} \lambda_i + \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j x_j x_i^T - \sum_j \sum_i \lambda_i \lambda_j y_i y_j x_i x_j^T \right]$

(no need to Remember)

$$\max_{\lambda} \left[\sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j x_j x_i^T \right]$$

The Lagrangian

So in svm we find λ's which maximizes above equation

What are the KKT conditions...

⇒ Solving this equation we get λ's

⇒ from λ's we can get $\omega \Rightarrow \sum_{i=1}^{N} \lambda_i y_i x_i$

The Lagrangian

we can see that Solution of λ depends on $x_j x_i^T \Rightarrow$ Single value

$$\begin{pmatrix} x_j^1 & x_j^2 & \cdots & x_j^D \end{pmatrix} \begin{bmatrix} x_i^1 \\ x_i^2 \\ \vdots \\ x_i^D \end{bmatrix}$$

- we will have many Pair
  N points $\Rightarrow N \times N \Rightarrow N^2$
  Pair

What are the KKT conditions...

SVM primal and dual

- Conclusion ⇒

1) Prob $\min \frac{1}{2}\|w\|^2$

   St $y_i(wx_i+b) \geq 1$

2) $w = \sum \lambda_i y_i x_i$

   ↳ governed by SV's

3) for solving λ's we need to find $x_i x_j^T$

SVM primal and dual

4) For converting the data into higher dimension @backend we use Kernel fxn

$$\left.\begin{array}{l} x_i \longrightarrow \phi(x_i) \\ x_j \longrightarrow \phi(x_j) \end{array}\right\}$$ no need of converting points into higher dimension

$\phi(x_i)\,\phi(x_j)^T \Rightarrow$ Can be written in terms of $K(x_i, x_j)$

SVM primal and dual

5) So the eq $\Rightarrow$

$$\Rightarrow \sum_{i=1}^{N} \lambda_i - \sum_i \sum_j \lambda_i \lambda_j y_i y_j x_i x_j^T$$

data not modified

$$x_i = (x_i^1, x_i^2, x_i^3)$$

$$x_j = (x_j, x_j^2, x_j^3)$$

$$f(x_i) = \{(x_i^1)^2, x_i^1 \cdot x_i^2, x_i^1 \cdot x_i^3, \; - - - - \; - \; - \; - \}$$

Convert into 9 dimension

$$\sum_{i=1}^{N} \lambda_i - \sum \sum \lambda_i \lambda_j y_i y_j f(x_i) f(x_j)^T$$

SVM primal and dual

$$\left[\sum \lambda_i - \sum \sum \lambda_i \lambda_j y_i y_j \underbrace{K(x_i \cdot x_j)}_{\text{Kernel fxn.}}\right]$$

SVM primal and dual

$$\left(\underbrace{x^1, x^2, x^3}, (x^1)^2, x^2 . x^1, (x^3)^2 \ldots\right)$$

So we have inc dimensionality of data

$$(x_i) \rightarrow \phi(x_i) \Rightarrow$$

$$(x^1, x^2, x^3) \longrightarrow (x^1, x^2, x^3, (x^1)^2, x_1 . x_2 \ldots)$$

How the SVMs classifier depend only on Support Vector

SVMs

Imp

SVM primal and dual

Simple Example: $x = (x1, x2, x3)$; $y = (y1, y2, y3)$. Then for the function $f(x) = (x1x1, x1x2, x1x3, x2x1, x2x2, x2x3, x3x1, x3x2, x3x3)$, the kernel is $K(x, y) = (\langle x, y\rangle)^2$.

Let's plug in some numbers to make this more intuitive: suppose $x = (1, 2, 3)$; $y = (4, 5, 6)$. Then:

$f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)$

$f(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)$

$\langle f(x), f(y)\rangle = 16 + 40 + 72 + 40 + 100 + 180 + 72 + 180 + 324 = 1024$

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

$K(x,y) = (\langle x \cdot y\rangle)^2$

$= 32^2 = 1024$

$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix} \Rightarrow 4 + 10 + 18 \Rightarrow 32$

Now let us use the kernel instead:

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

Same result, but this calculation is so much easier.

How the SVMs classifier depend only on Support Vector

⇒ In SVM dual we need $(x_i x_j^T)$ ⇒ we do not need points in higher dimension

we need inner product of the points

Let use a function i.e kernel

$$K(x,y) \Rightarrow \langle f(x) \cdot f(y) \rangle$$

SVMs

Imp

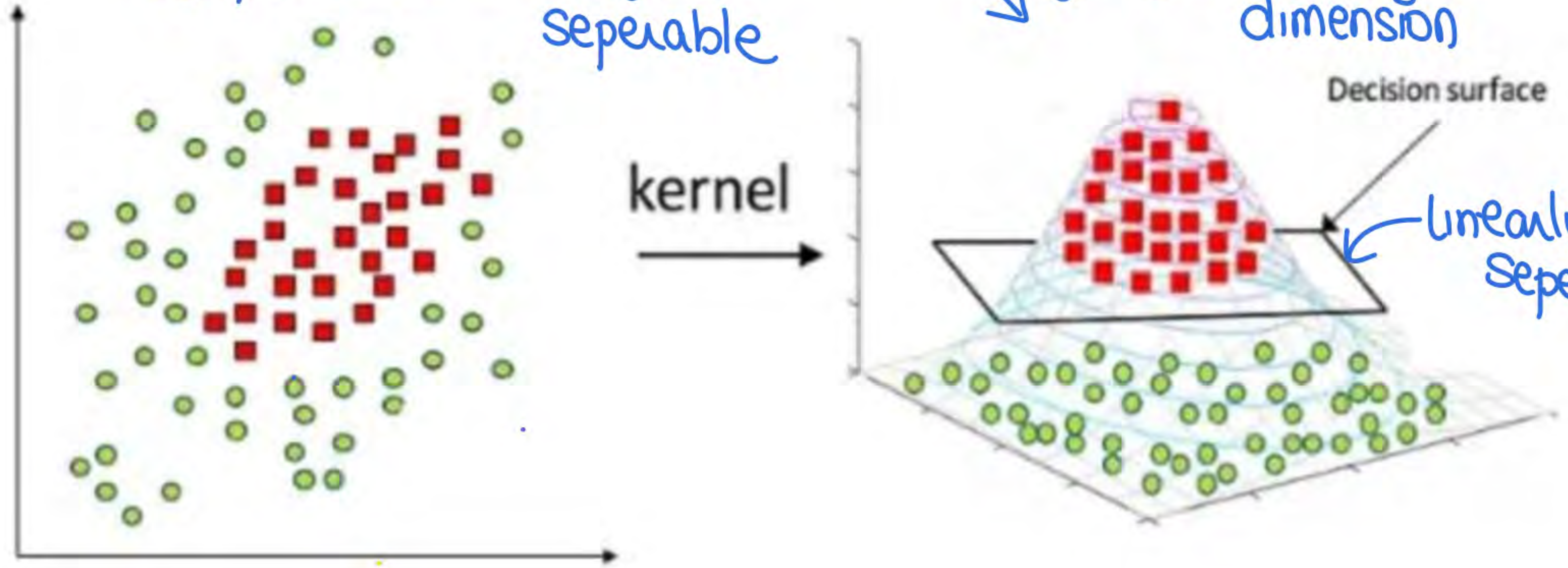
Use of kernels in SVM

- The "Kernel Trick" is a method used in Support Vector Machines (SVMs) to convert data (that is not linearly separable) into a higher-dimensional feature space where it may be linearly separated.
- This technique enables the SVM to identify a hyperplane that separates the data with the maximum margin, even when the data is not linearly separable in its original space. The kernel functions are used to compute the inner product between pairs of points in the transformed feature space without explicitly computing the transformation itself. This makes it computationally efficient to deal with high dimensional feature spaces.

Use of kernels in SVM

2D points not linearly seperable

going to higher dimension



linearly seperable

Here are some most commonly used kernel functions in SVMs:

The linear kernel can be defined as:

$$K(x, y) = x . y$$

Here are some most commonly used kernel functions in SVMs:

One definition of the polynomial kernel is:

Where x and y are the input feature vectors, c is a constant term, and d is the degree of the polynomial, K(x, y) = (x. y + c)d. The constant term is added to, and the dot product of the input vectors elevated to the degree of the polynomial.

Here are some most commonly used kernel functions in SVMs:

The Gaussian kernel can be defined as:

```
K(x, y) = exp(-gamma * ||x - y||^2)
```

Here are some most commonly used kernel functions in SVMs:

The Laplacian kernel can be defined as:

```
K(x, y) = exp(-gamma * ||x - y||)
```

THANK - YOU