CLASSIFICATION

WHAT WE SAW YESTERDAY?

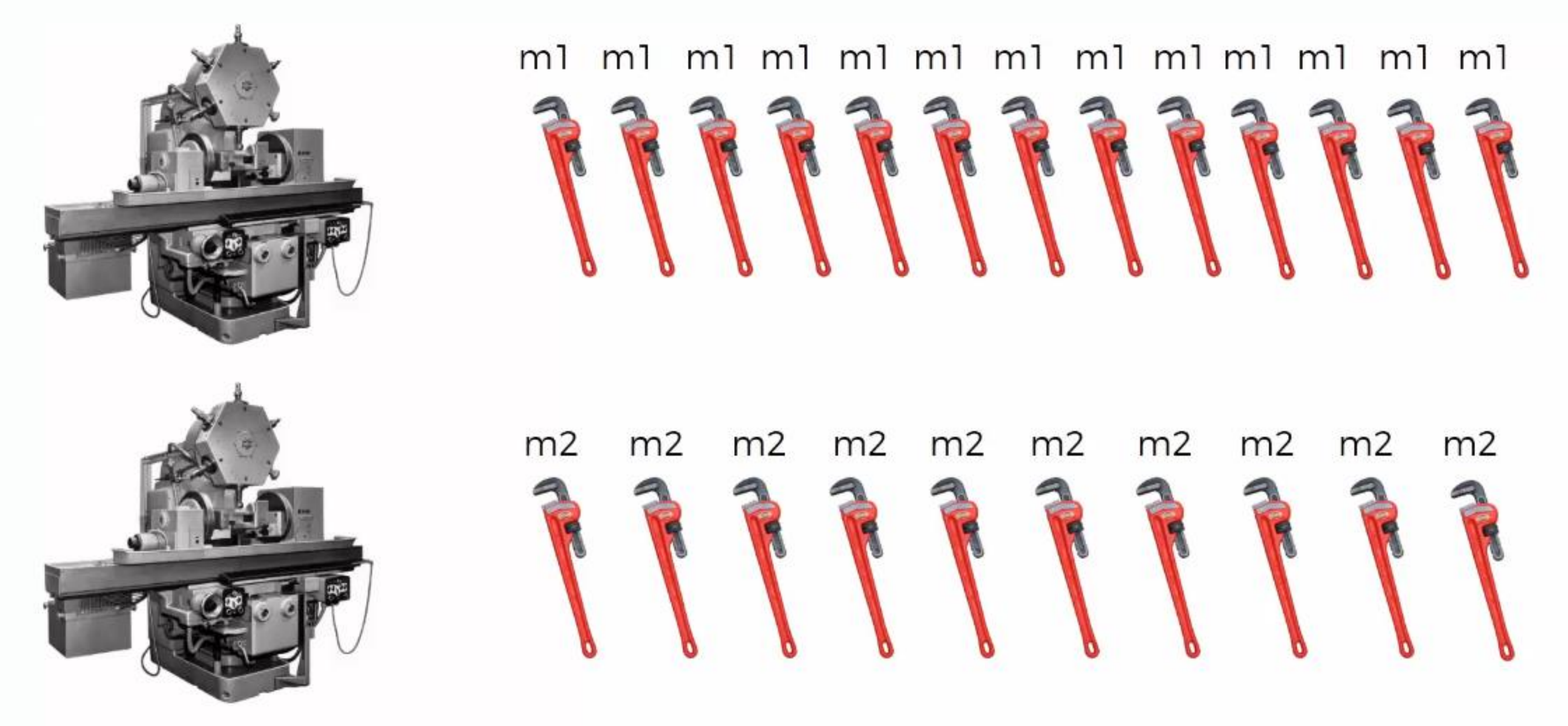
- Logistic Regression
- K-Nearest Neighbours

AGENDA FOR TODAY

- Naïve Bayes
- Decision Trees
- Random Forest
- Support Vectors

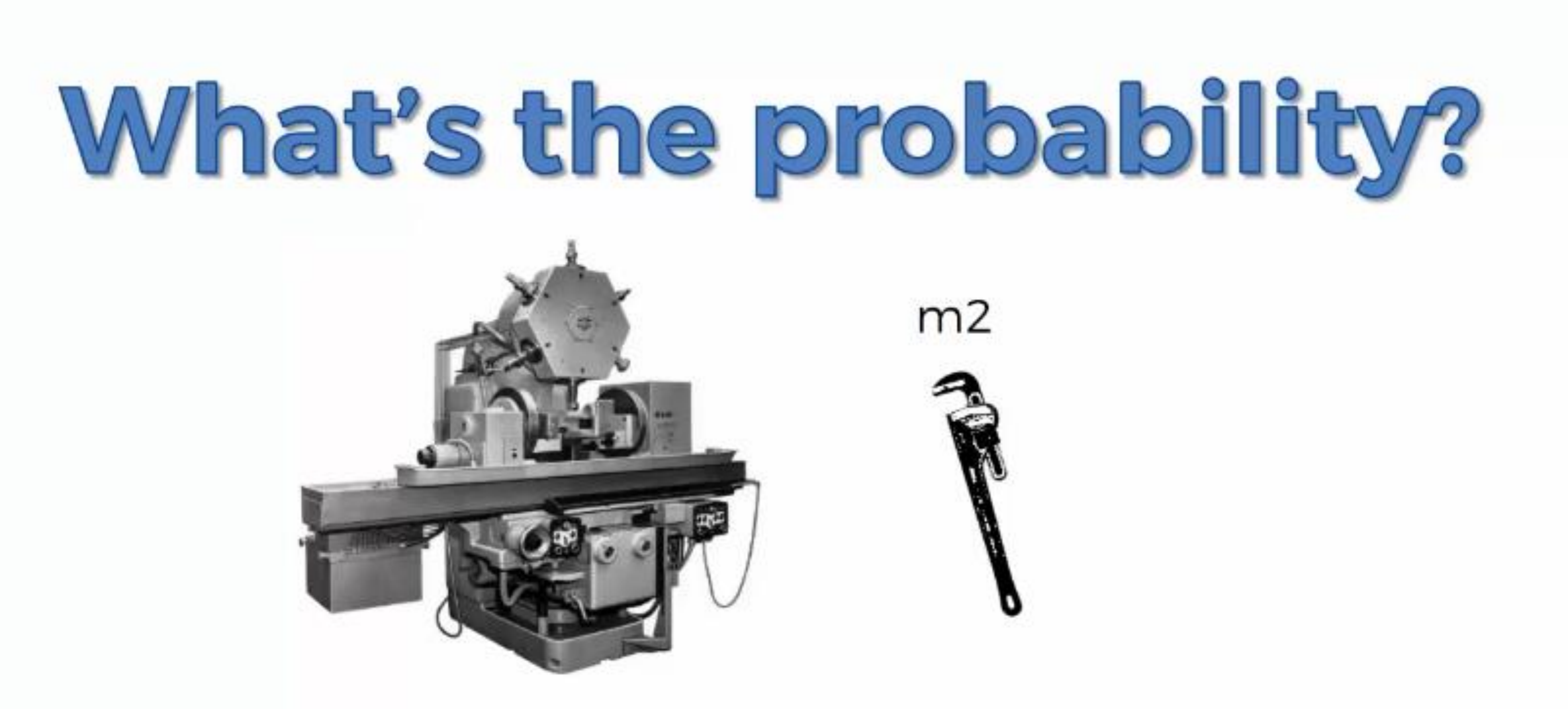
NAÏVE BAYES

BAYES THEOREM


m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1 m1
m2 m2 m2 m2 m2 m2 m2 m2 m2 m2

FIND DEFECTIVE SPANNERS FROM M2

Objective: Find the probability of a spanner picked from the lot to be defective spanner from M2



BAYES THEOREM

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

EXAMPLE:

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

EXAMPLE

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

Out of all produced parts:
We can SEE that 1% are defective

EXAMPLE

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

Out of all produced parts:
We can SEE that 1% are defective

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

Out of all produced parts:
We can SEE that 1% are defective

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

Question:
What is the probability that a part produced by mach2 is defective = ?

SOLUTION

WHAT IS GIVEN?

Mach1: 30 wrenches / hr -> P(Mach1) = 30/50 = 0.6

Mach2: 20 wrenches / hr -> P(Mach2) = 20/50 = 0.4

WHAT IS GIVEN?

Out of all produced parts:
We can SEE that 1% are defective -> P(Defect) = 1%

WHAT IS GIVEN?

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

-> P(Mach1 | Defect) = 50%
-> P(Mach2 | Defect) = 50%

WHAT IS GIVEN?

Mach1: 30 wrenches / hr -> P(Mach1) = 30/50 = 0.6

Mach2: 20 wrenches / hr -> P(Mach2) = 20/50 = 0.4

Out of all produced parts:
We can SEE that 1% are defective -> P(Defect) = 1%

Out of all defective parts:
We can SEE that 50% came from mach1 -> P(Mach1 | Defect) = 50%
And 50% came from mach2 -> P(Mach2 | Defect) = 50%

OBJECTIVE IS TO FIND P(DEFECT|M2)?

USING BAYES THEOREM:

$$P(Defect \mid Mach2) = \frac{P(Mach2 \mid Defect) * P(Defect)}{P(Mach2)}$$

$$P(Defect \mid Mach2) = \frac{0.5 \quad * \quad 0.01}{0.4} = 0.0125$$

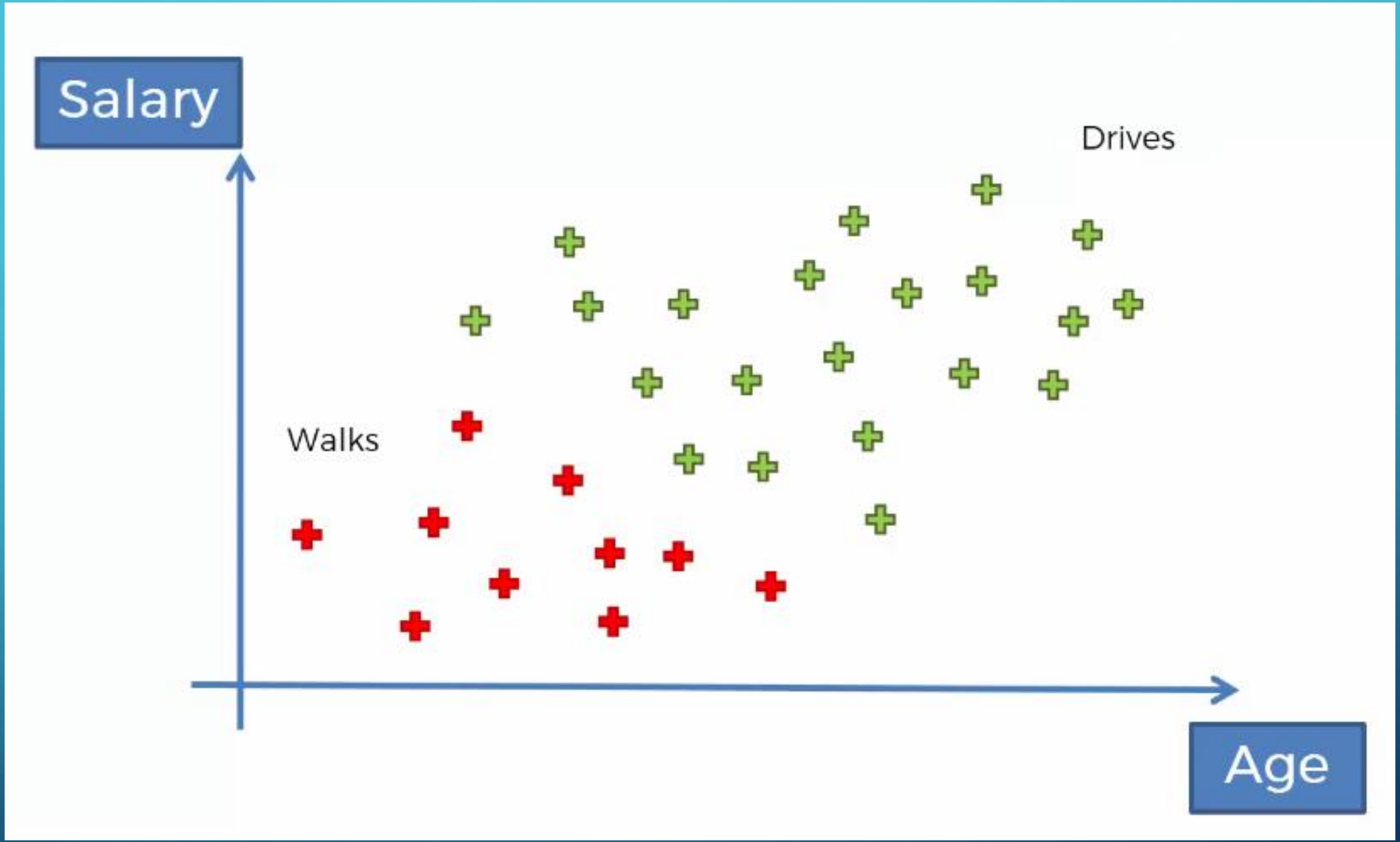
ITS INTUITIVE!!

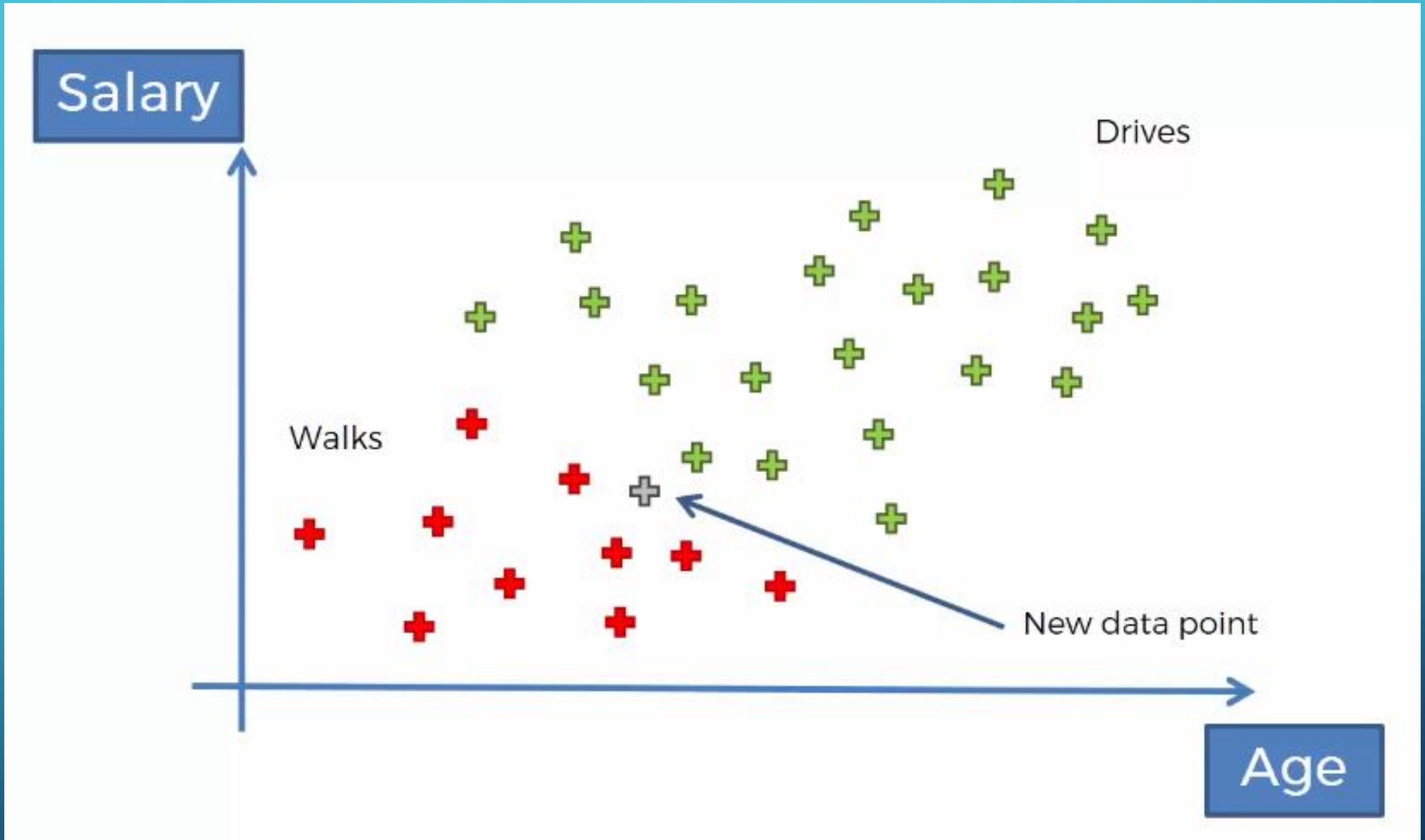
<u>**Let's look at an example:**</u>

- 1000 wrenches
- 400 came from Mach2
- 1% have a defect = 10
- of them 50% came from Mach2 = 5
- % defective parts from Mach2 = 5/400 = 1.25%

NAÏVE BAYES CLASSIFIER

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$


Salary
Drives
Walks
Age


Salary
Drives
Walks
New data point
Age

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

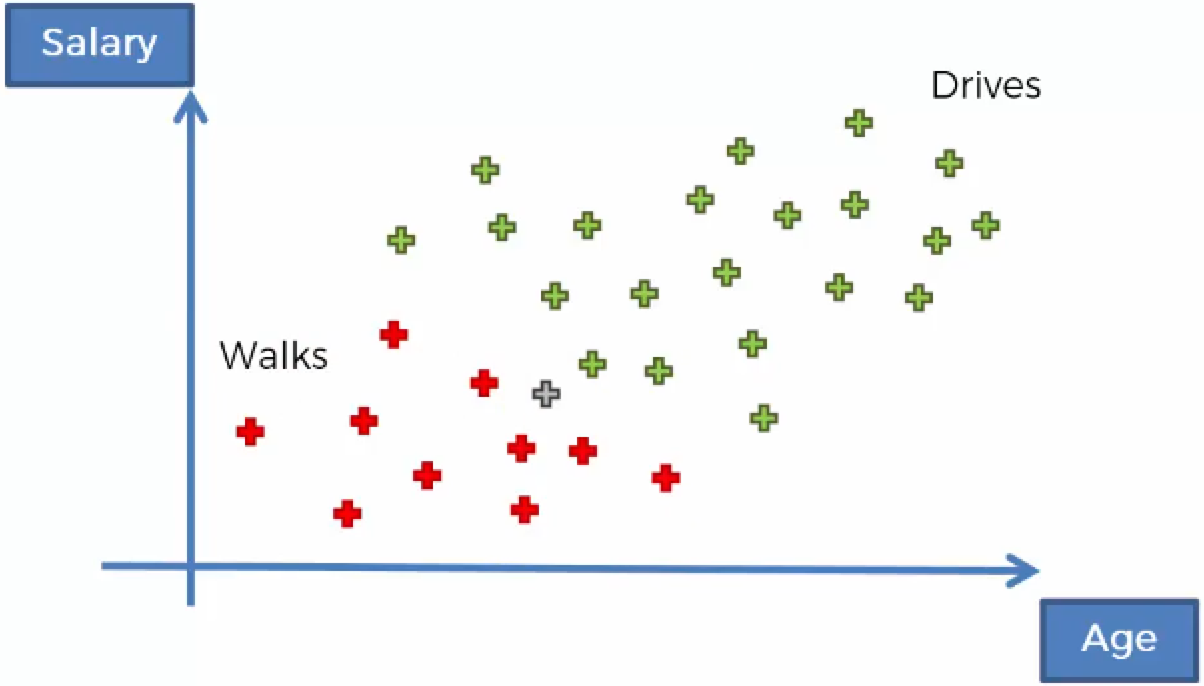
#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{P(X|Walks) * P(Walks)}{P(X)}$$

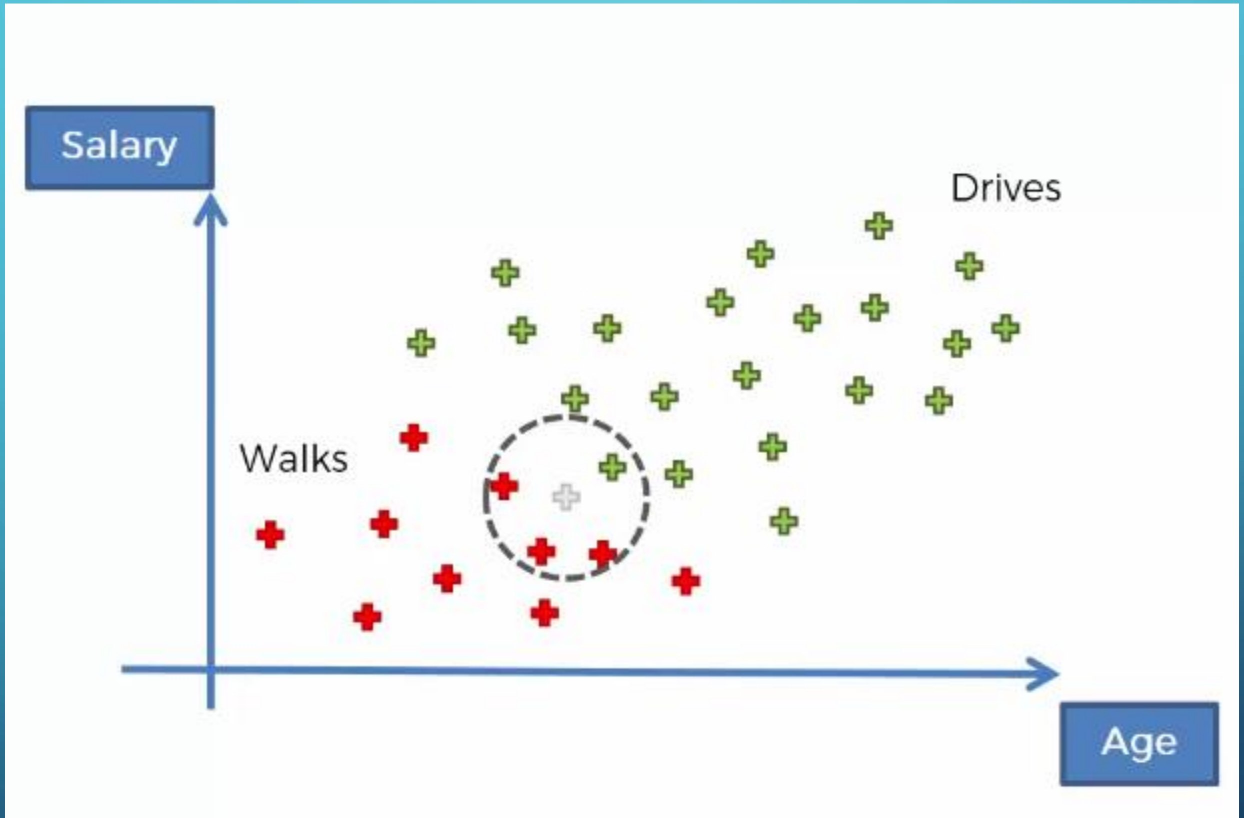
#2 Marginal Likelihood

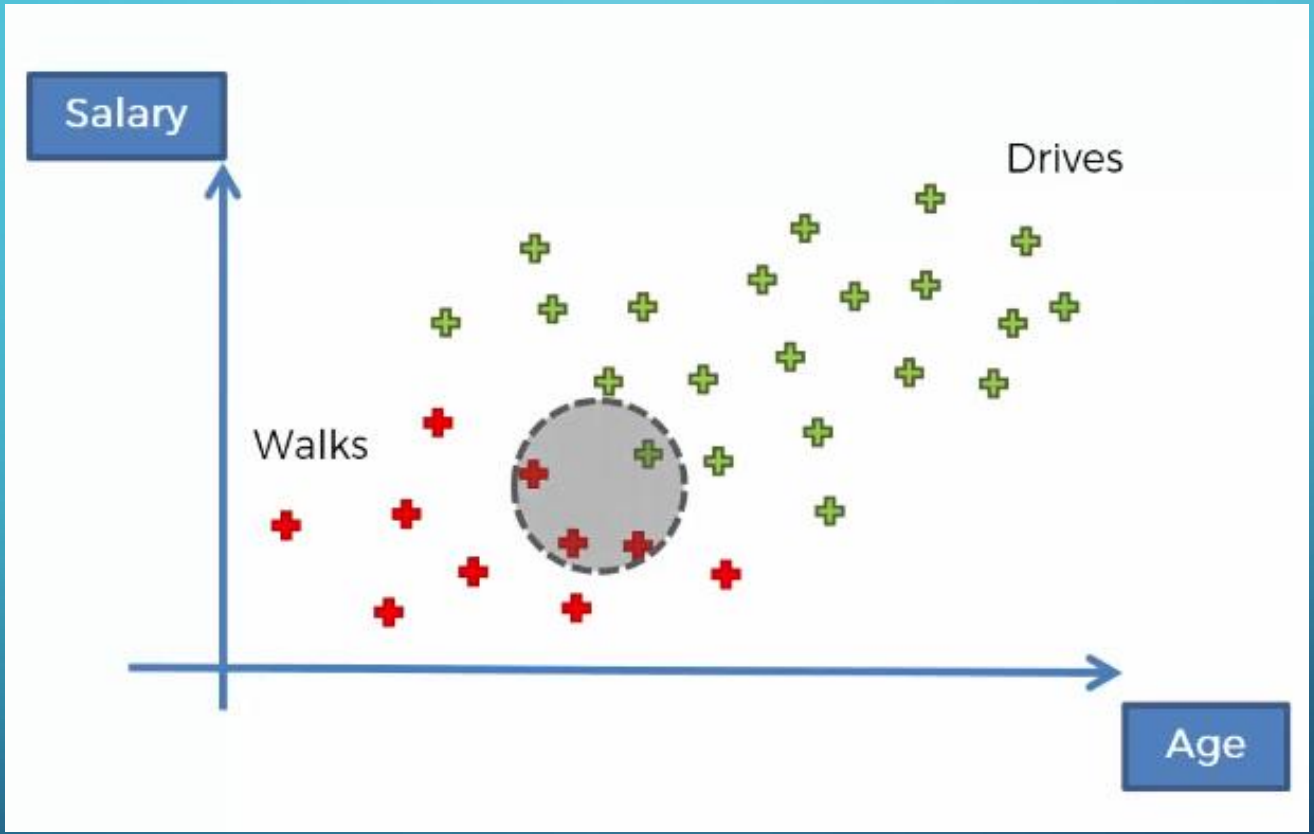


#1. P(Walks)

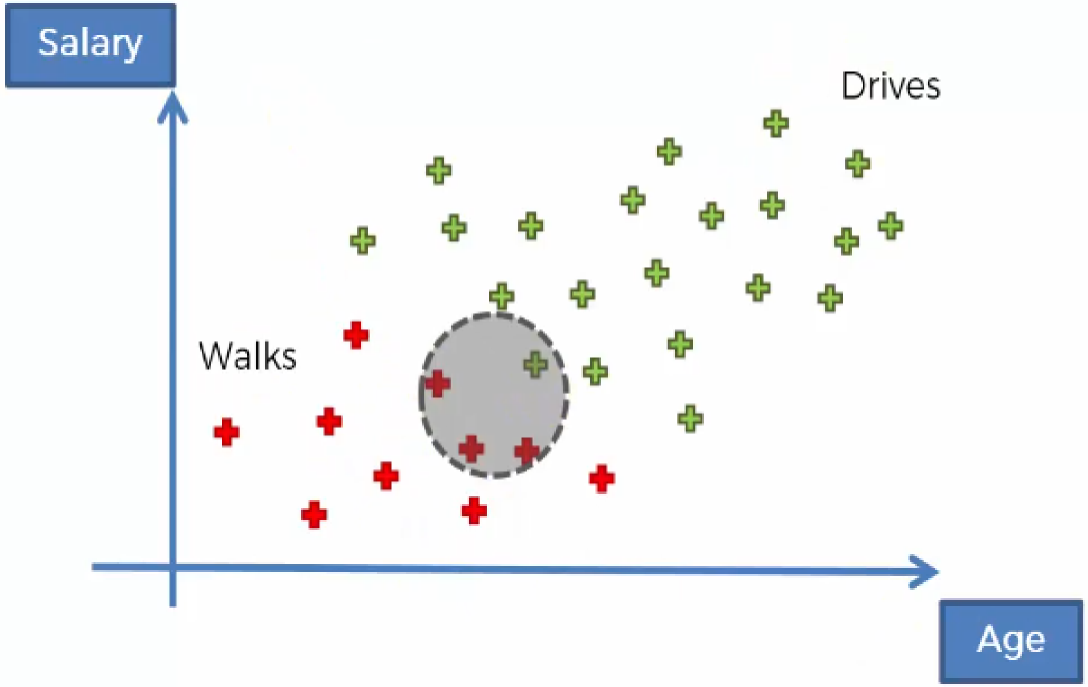
$$P(Walks) = \frac{Number\ of\ Walkers}{Total\ Observations}$$

FIND P(X):





Salary
Drives
Walks
Age

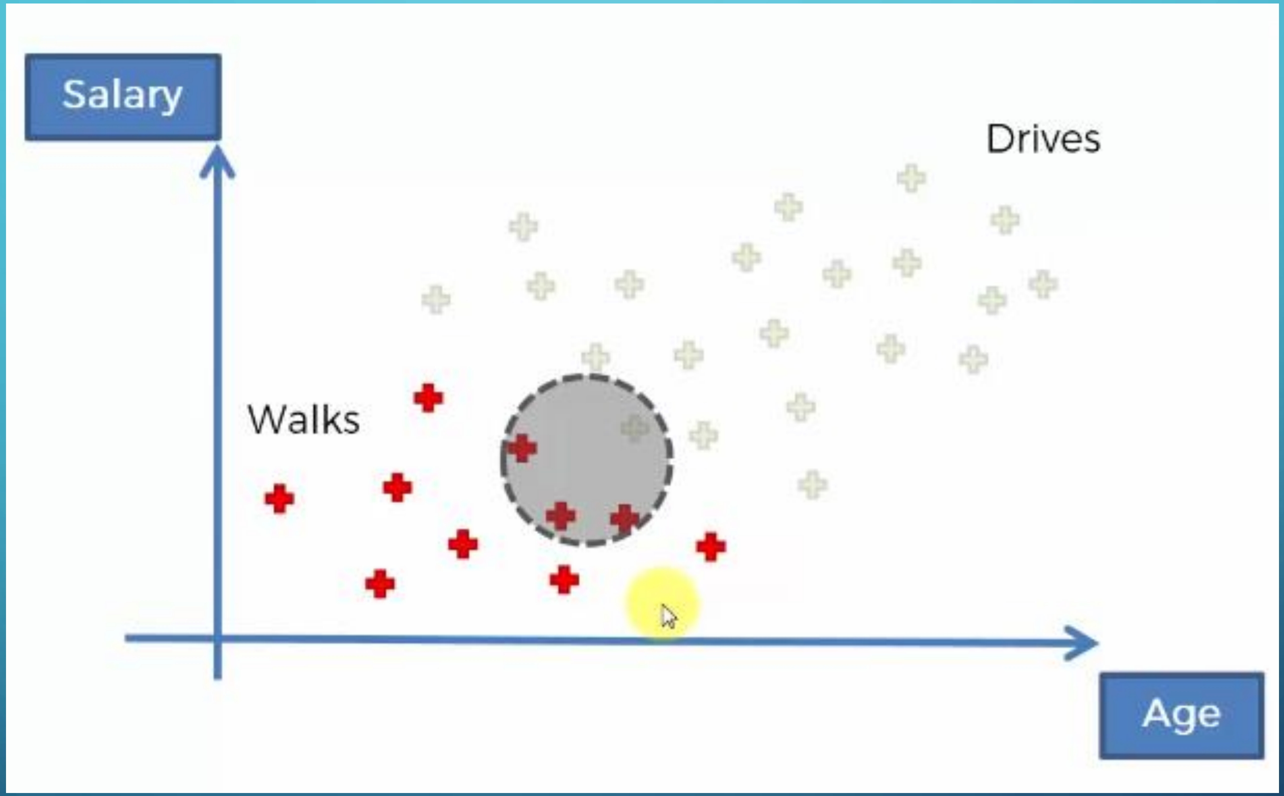


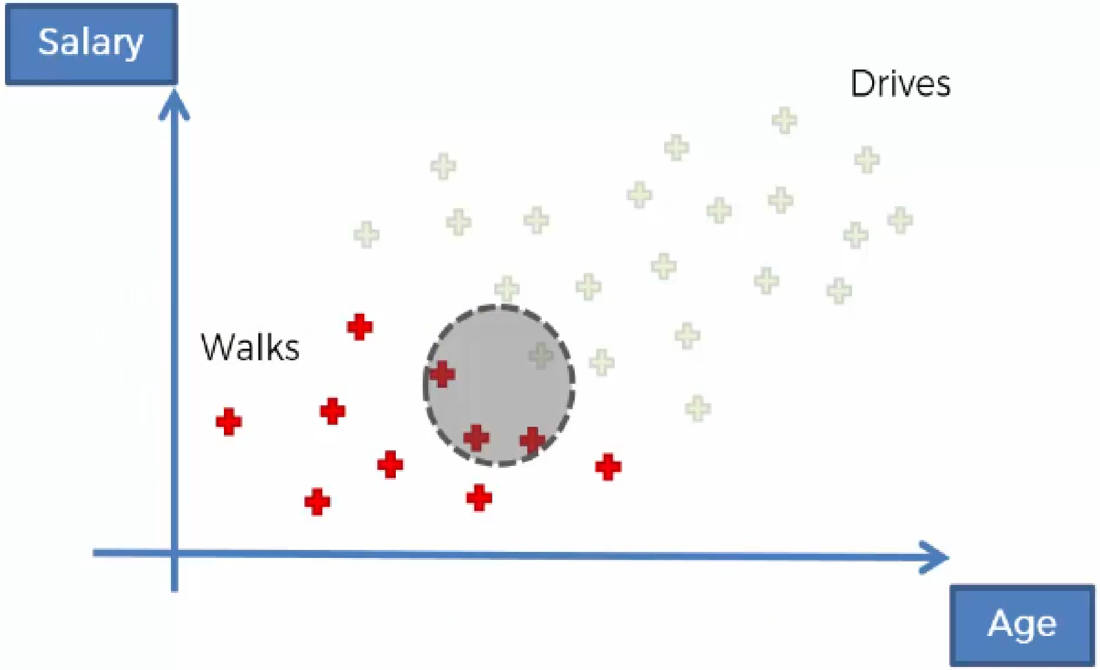
#2. P(X)

$$P(X) = \frac{Number\ of\ Similar\ Observations}{Total\ Observations}$$

$$P(X) = \frac{4}{30}$$

FIND P(X | WALKS)





#3. P(X|Walks)

$$P(X|Walks) = \frac{\text{Number of Similar Observations Among those who Walk}}{\text{Total number of Walkers}}$$

$$P(X|Walks) = \frac{3}{10}$$

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Walks|X) = \frac{\frac{3}{10} * \frac{10}{30}}{\frac{4}{30}} = 0.75$$

#2 Marginal Likelihood

SIMILARLY:

#4 Posterior Probability

#3 Likelihood

#1 Prior Probability

$$P(Drives|X) = \frac{P(X|Drives) * P(Drives)}{P(X)}$$

#2 Marginal Likelihood

P(DRIVES|X)

$P(Drives|X) = 1 - P(Walks|X)$ …. Bcoz there are two classes

$$0.75 \ v.s. \ 0.25$$

$$P(Walks|X) > P(Drives|X)$$


Salary
Drives
Walks
New data point
Age

Additional Comments:

1. Why Naïve?

2. P(X)

3. More than 2 classes

Q: Why “Naïve”?

A: Independence assumption

2. P(X)

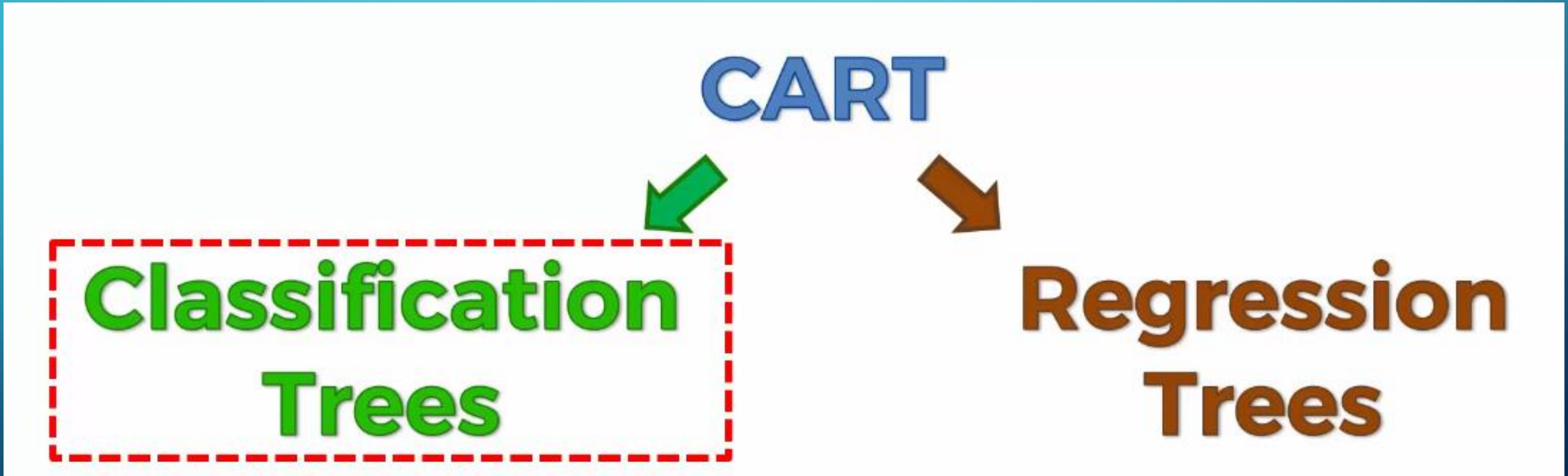
$$\frac{P(X|Walks) * P(Walks)}{\cancel{P(X)}} \quad v.s. \quad \frac{P(X|Drives) * P(Drives)}{\cancel{P(X)}}$$

3. MULTIPLE CLASSES

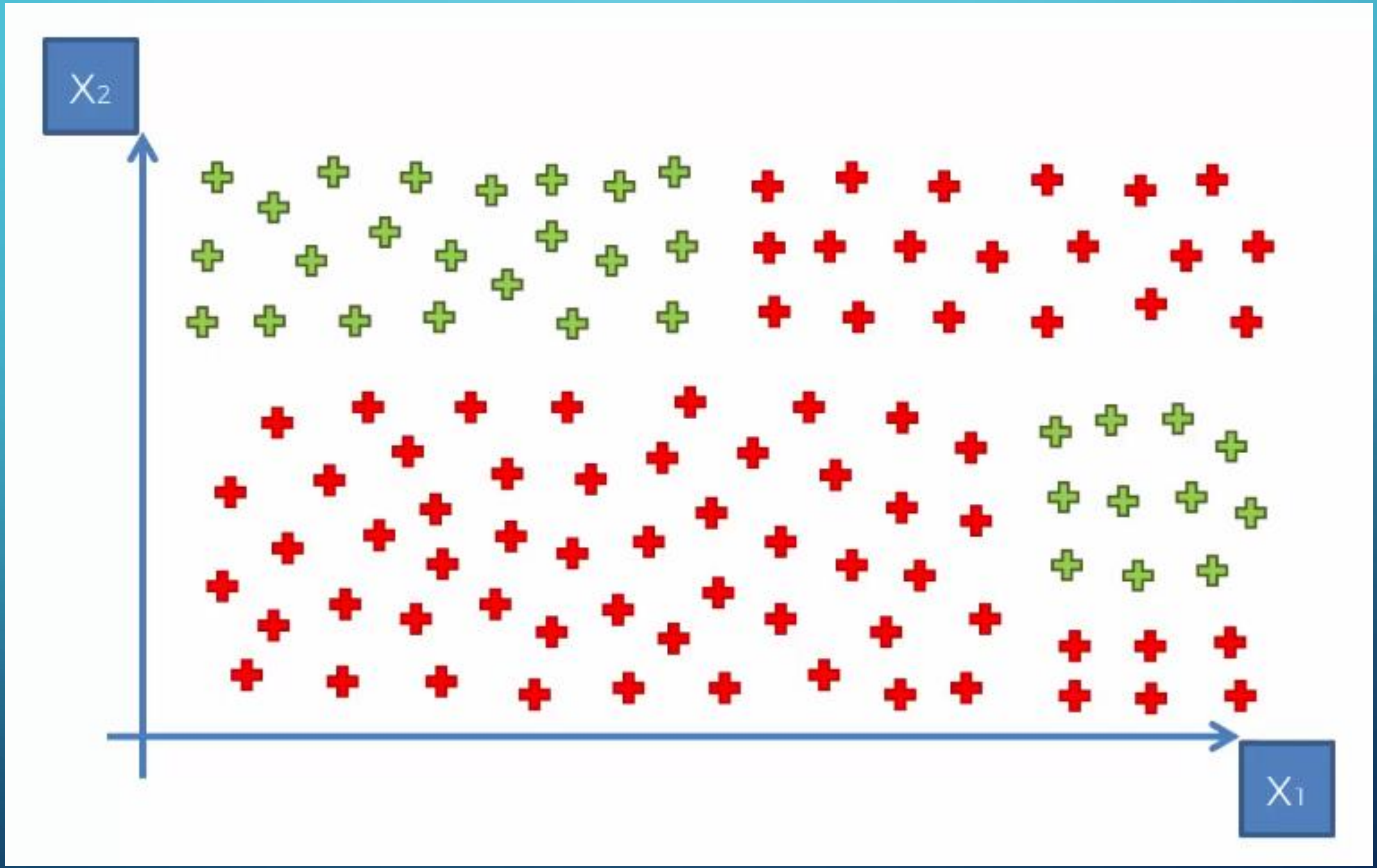
- Earlier, we had calculated for two. Now we would have to do the same for other classes as well.
- Eg: P(Walks|X) vs P(Drives|X) vs P(Runs|X). Then max probability class will be considered.

DECISION TREES

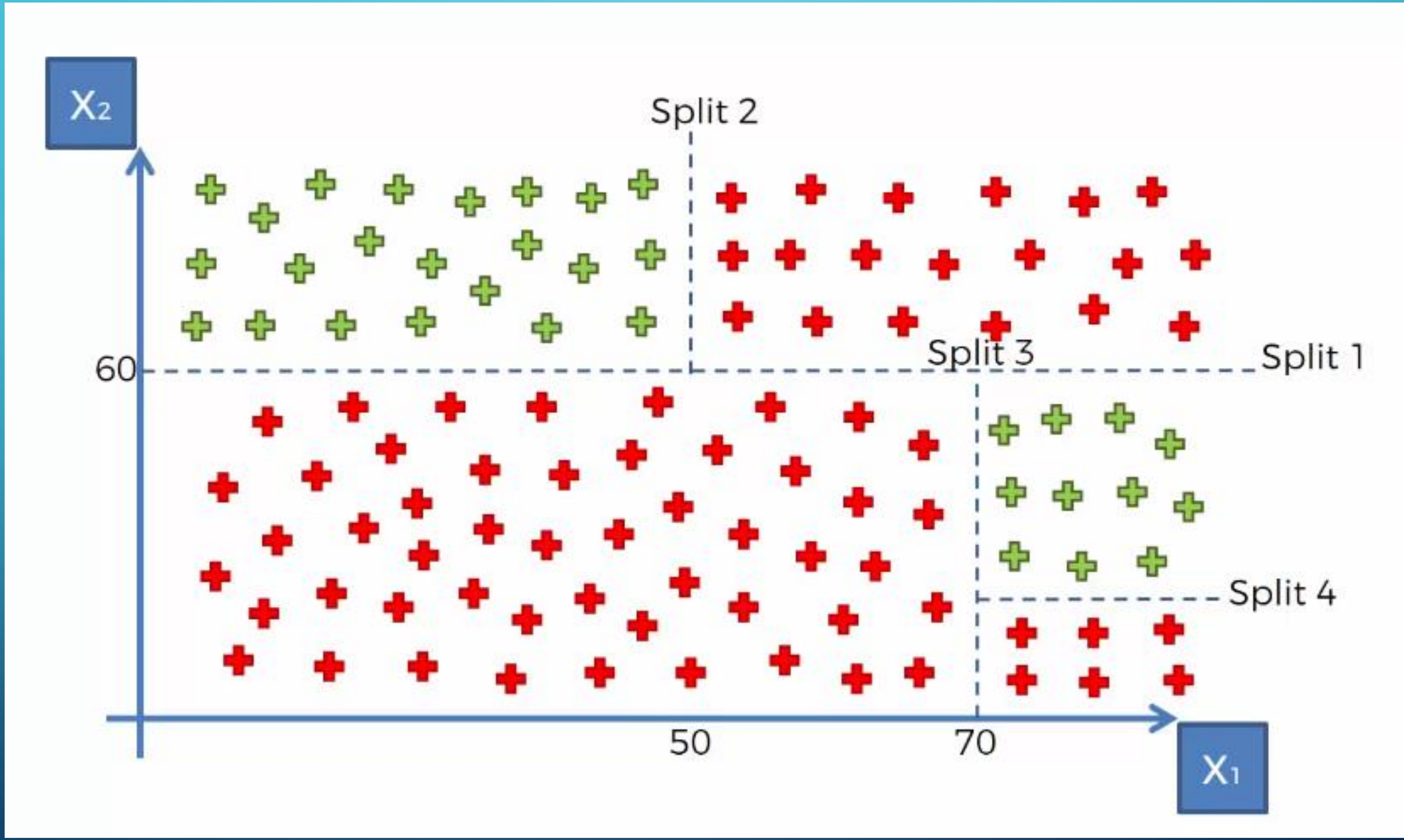
CLASSIFICATION AND REGRESSION TREES



EXAMPLE



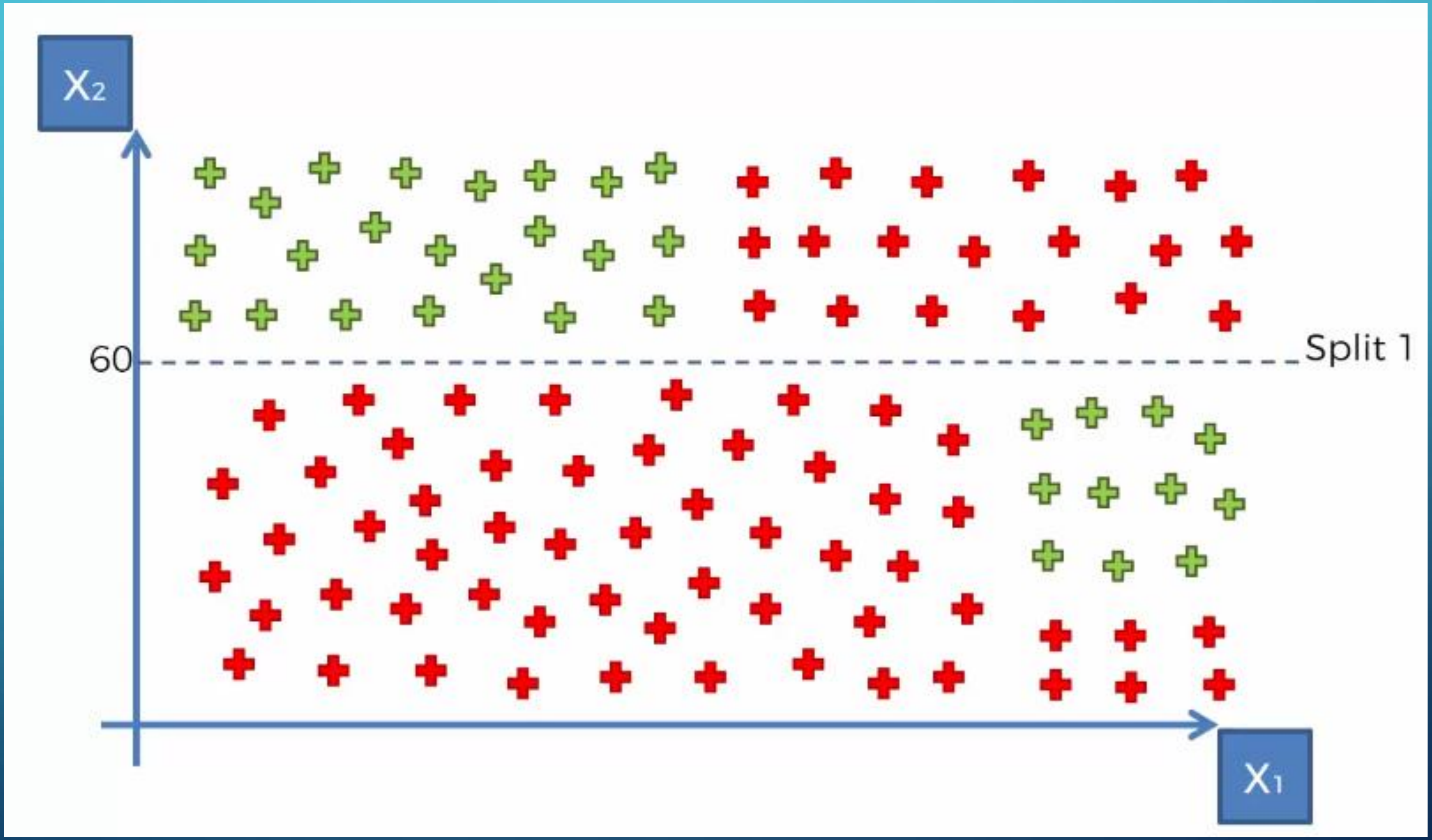
LEAVES OF DECISION TREE



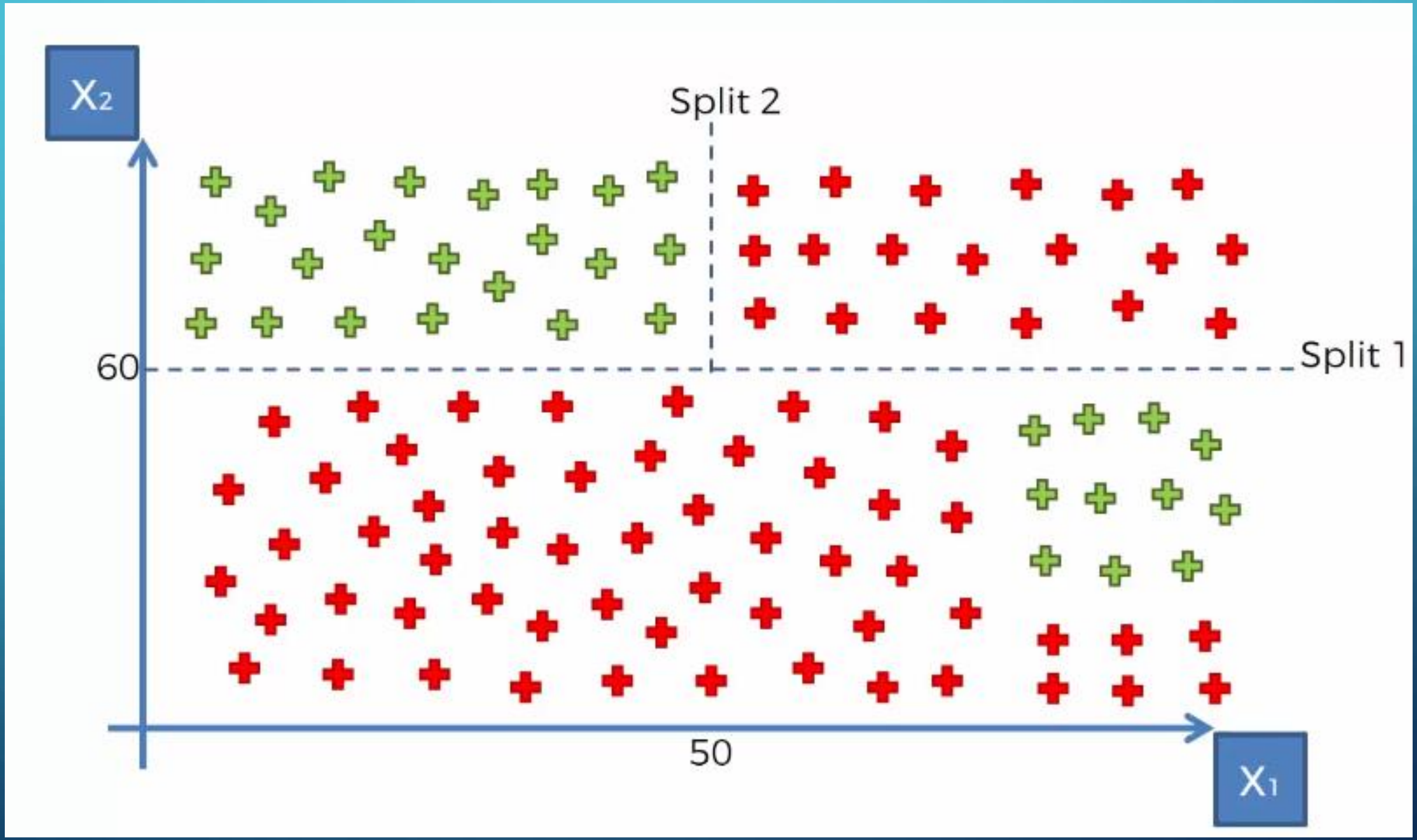
HOW IS IT CREATED??

- Using Information Entropy ----- Out of Scope

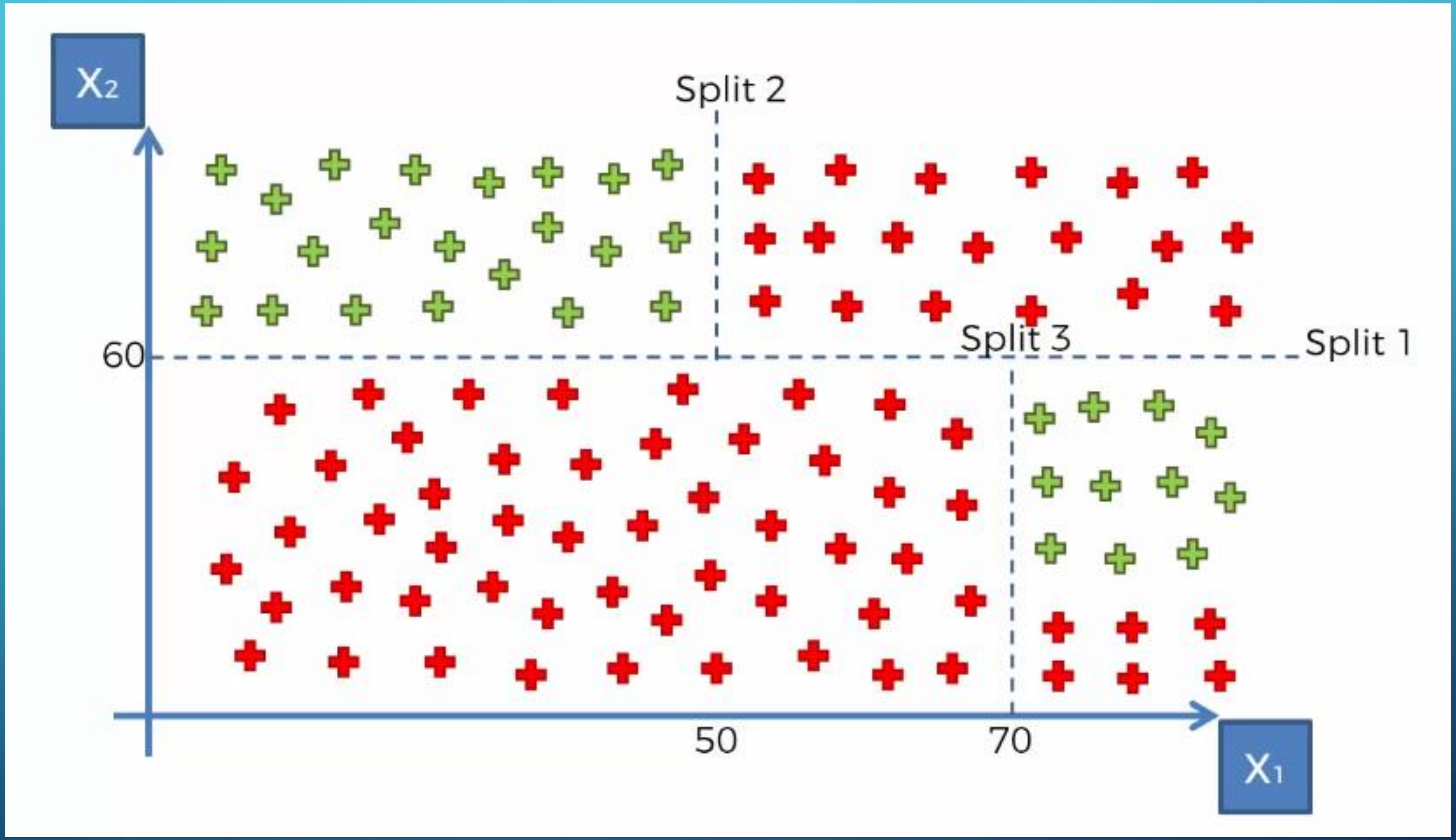
LETS SEE HOW LEAVES ARE CREATED…



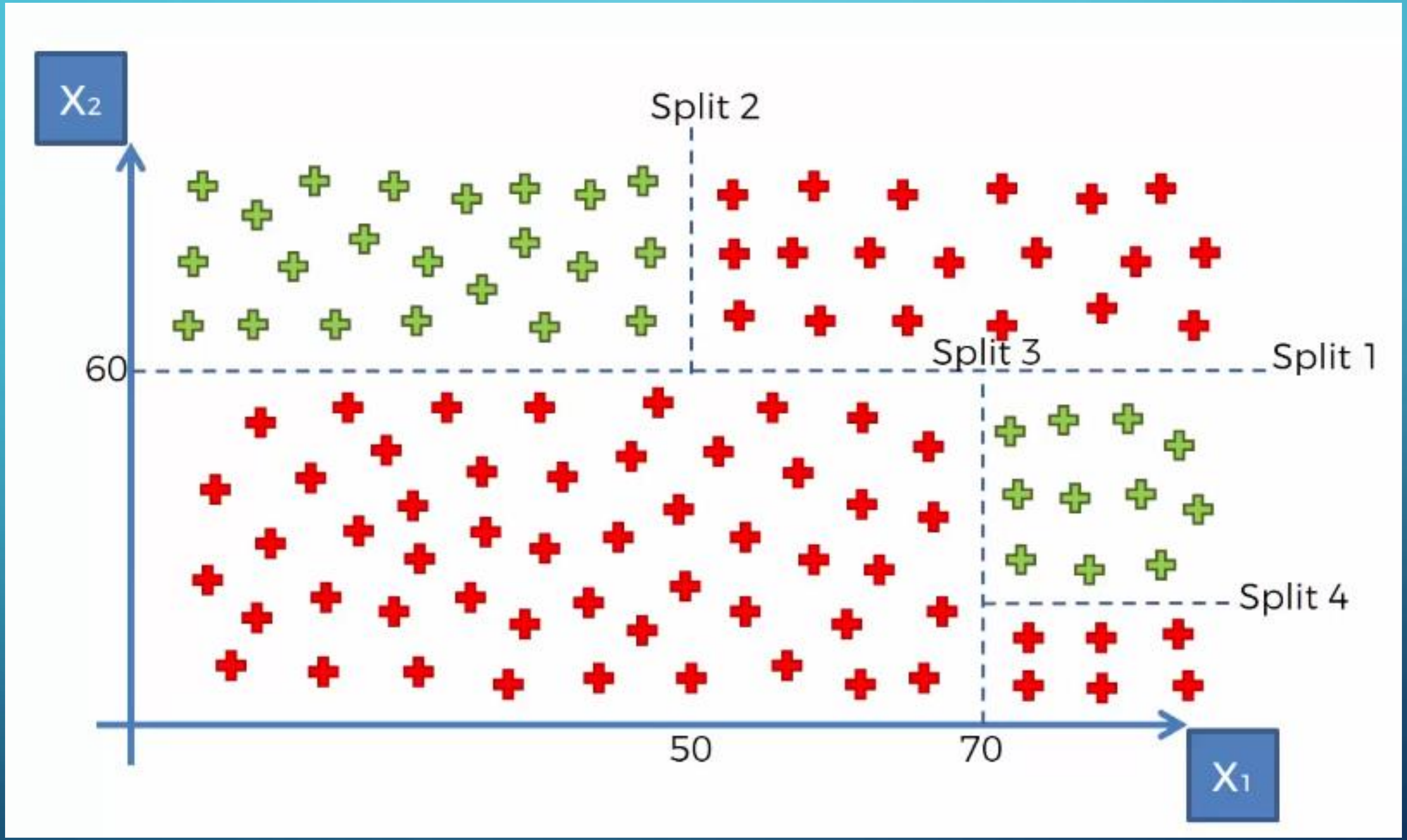
2ND SPLIT:



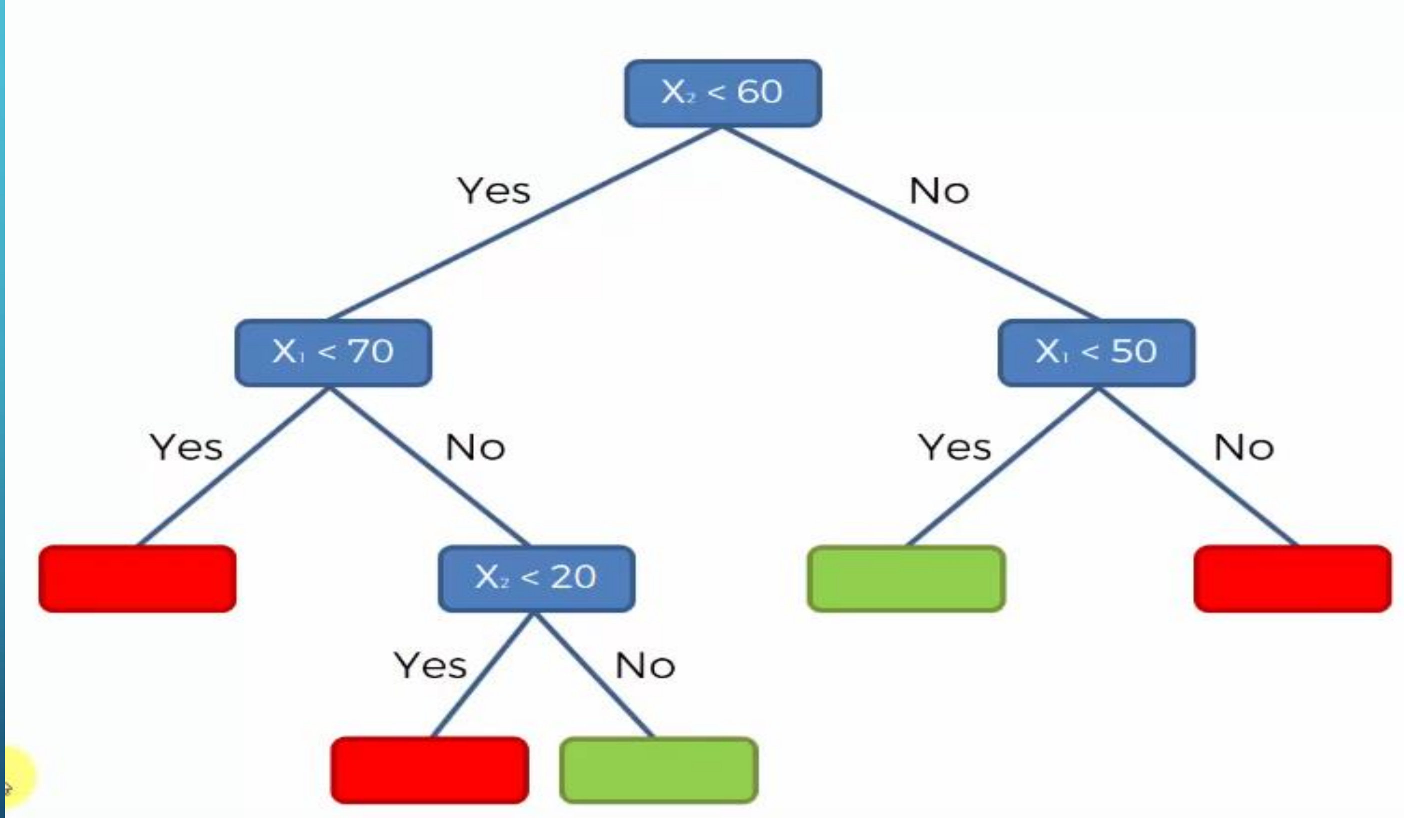
3RD SPLIT



4TH SPLIT:



TREE DIAGRAM:



SUMMARY:

- Old Method and almost got extinct… lol!!
- Reborn with upgrades and now used in major algorithms like Random Forest

RANDOM FOREST

WHAT IS ENSEMBLE LEARNING?

- Ensemble Learning uses multiple learning algorithms to obtain better predictive performance

RANDOM FOREST INTUITION

STEP 1: Pick at random K data points from the Training set.

STEP 2: Build the Decision Tree associated to these K data points.

STEP 3: Choose the number Ntree of trees you want to build and repeat STEPS 1 & 2

STEP 4: For a new data point, make each one of your Ntree trees predict the category to which the data points belongs, and assign the new data point to the category that wins the majority vote.

XBOX GAME KINECT

https://www.i-programmer.info/news/105-artificial-intelligence/2176-kinects-ai-breakthrough-explained.html

Real-Time Human Pose Recognition in Parts from Single Depth Images

Jamie Shotton Andrew Fitzgibbon Mat Cook Toby Sharp Mark Finocchio
Richard Moore Alex Kipman Andrew Blake
Microsoft Research Cambridge & Xbox Incubation

Abstract

We propose a new method to quickly and accurately predict 3D positions of body joints from a single depth image, using no temporal information. We take an object recognition approach, designing an intermediate body parts representation that maps the difficult pose estimation problem into a simpler per-pixel classification problem. Our large and highly varied training dataset allows the classifier to estimate body parts invariant to pose, body shape, clothing, etc. Finally we generate confidence-scored 3D proposals of several body joints by reprojecting the classification result and finding local modes.

The system runs at 200 frames per second on consumer hardware. Our evaluation shows high accuracy on both synthetic and real test sets, and investigates the effect of several training parameters. We achieve state of the art accuracy in our comparison with related work and demonstrate improved generalization over exact whole-skeleton nearest neighbor matching.

1. Introduction

Robust interactive human body tracking has applications including gaming, human-computer interaction, security, telepresence, and even health-care. The task has recently been greatly simplified by the introduction of real-

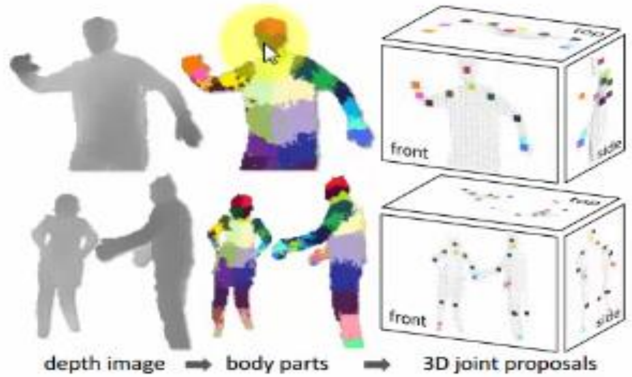


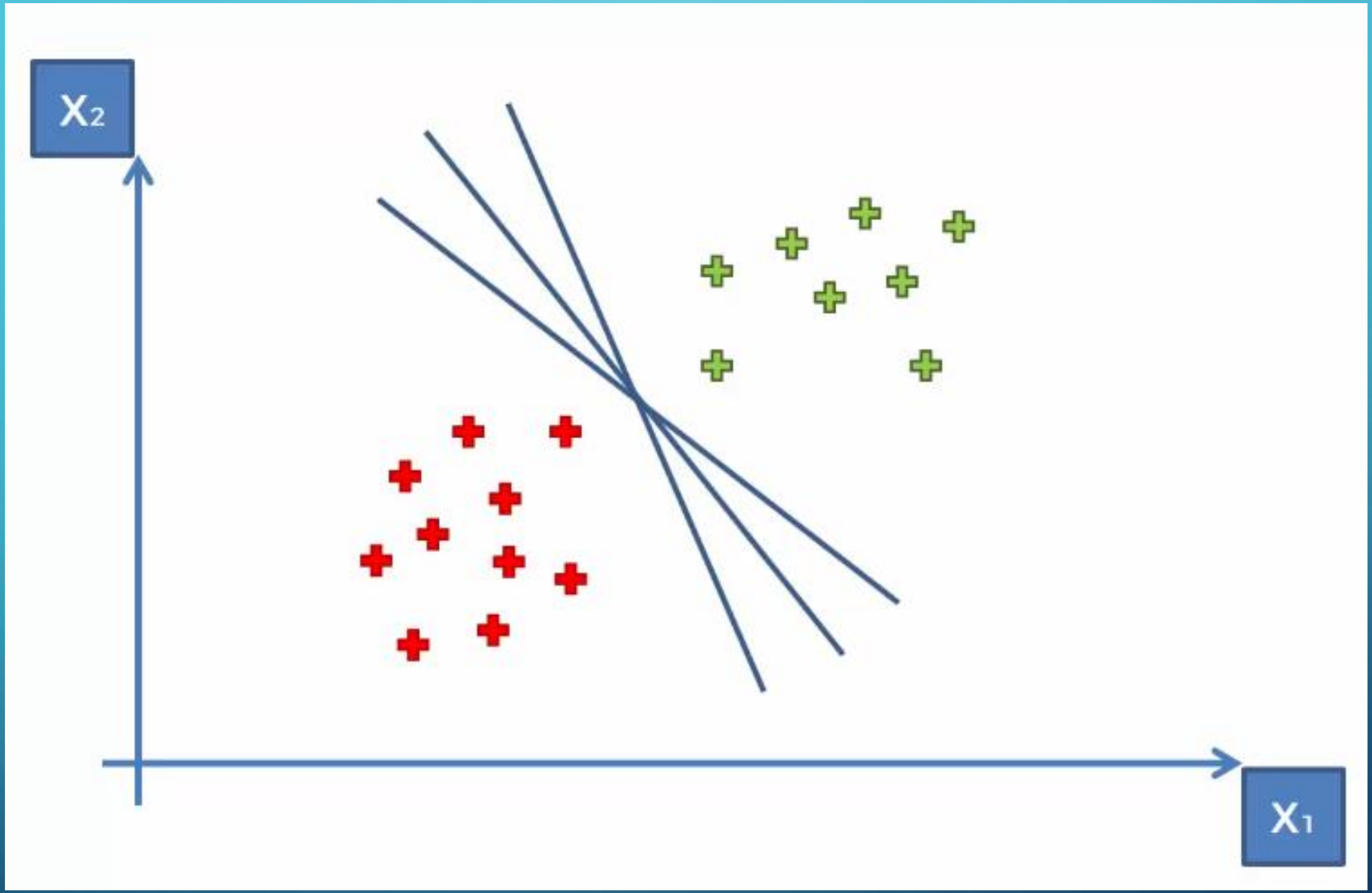
Figure 1. **Overview.** From an single input depth image, a per-pixel body part distribution is inferred. (Colors indicate the most likely part labels at each pixel, and correspond in the joint proposals). Local modes of this signal are estimated to give high-quality proposals for the 3D locations of body joints, even for multiple users.

joints of interest. Reprojecting the inferred parts into world space, we localize spatial modes of each part distribution and thus generate (possibly several) confidence-weighted proposals for the 3D locations of each skeletal joint.

We treat the segmentation into body parts as a per-pixel classification task (no pairwise terms or CRF have proved

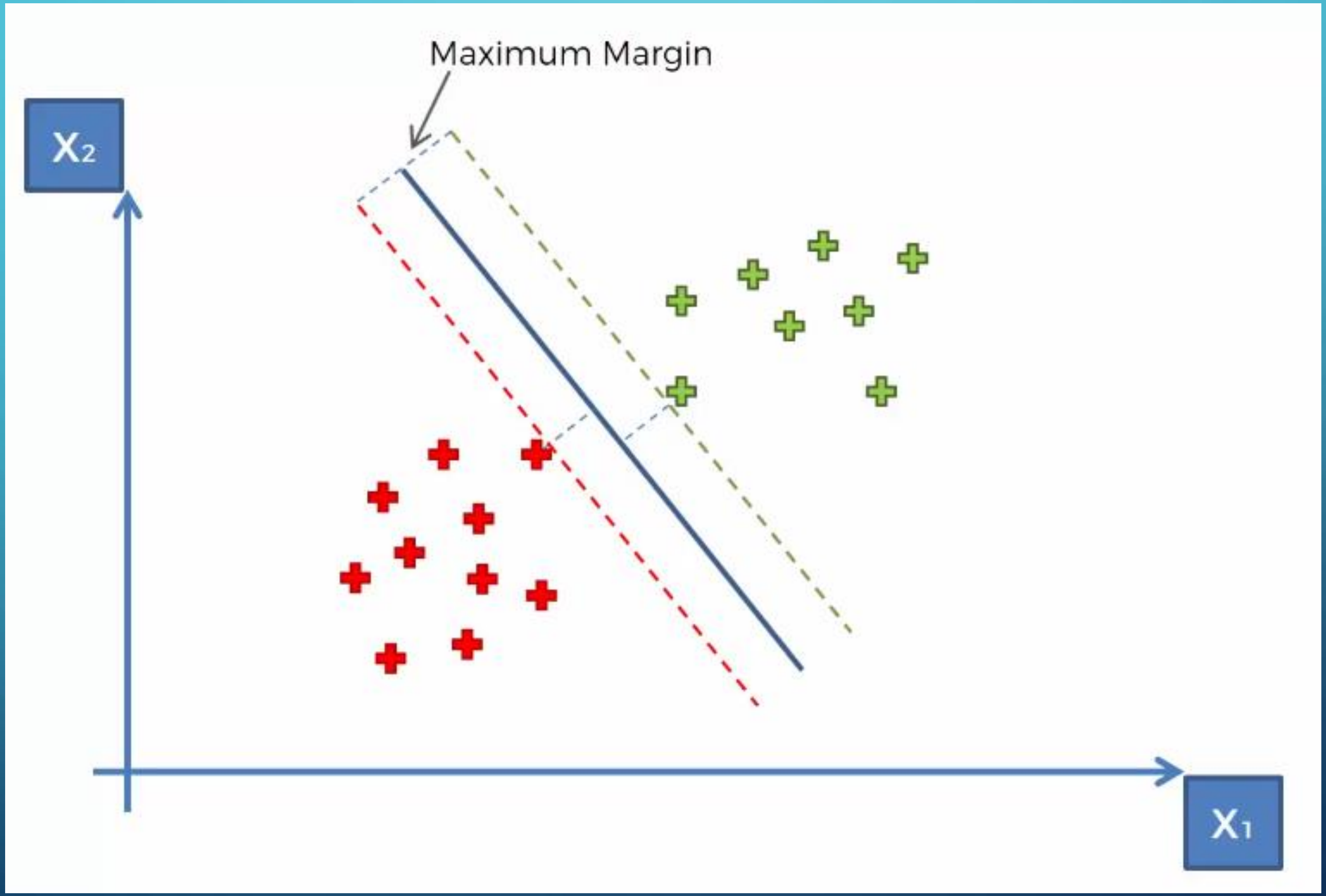
SUPPORT VECTOR MACHINE

SVM INTUITION

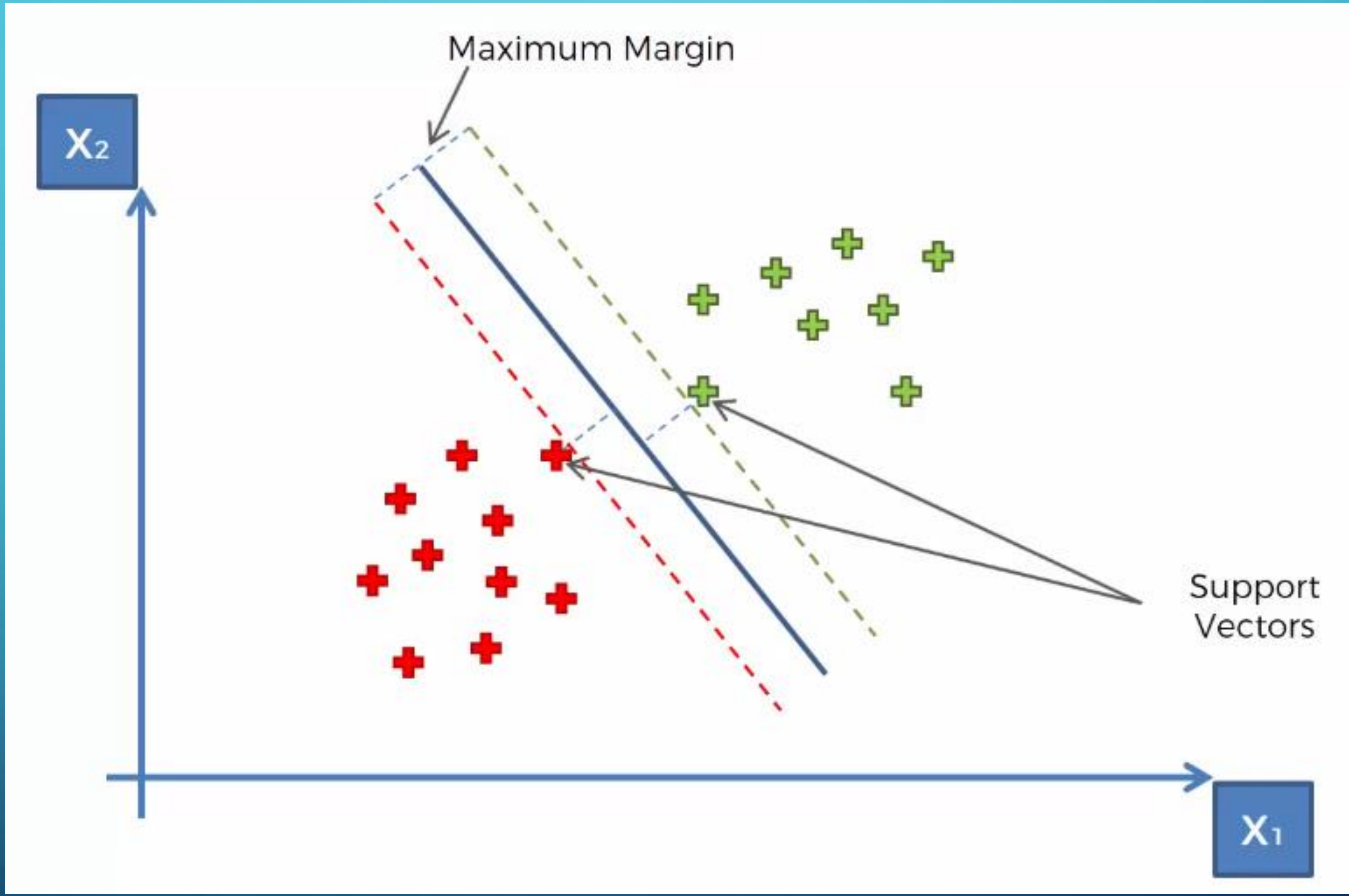


X2
X1

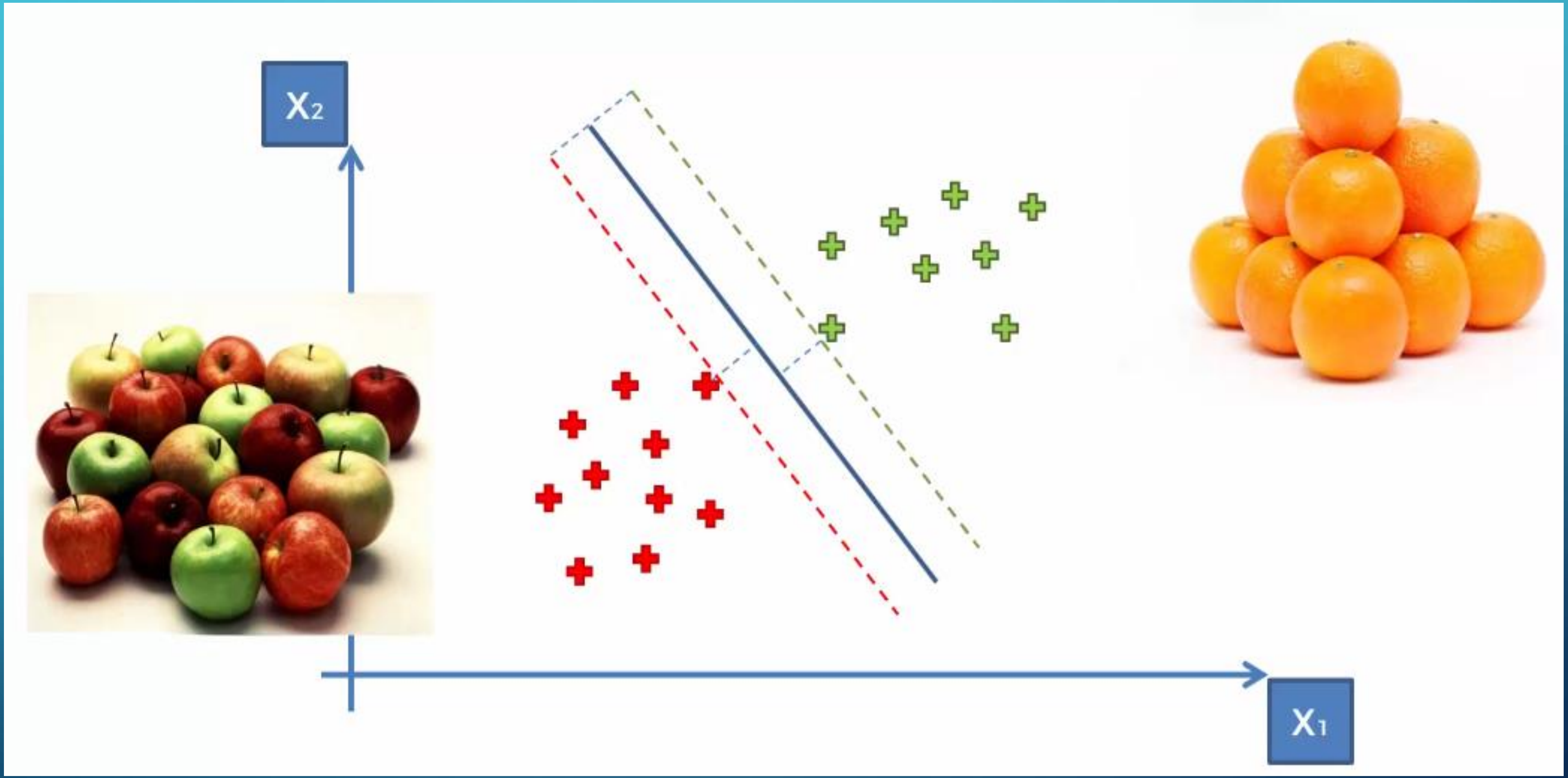
MAXIMUM MARGIN HYPERPLANE

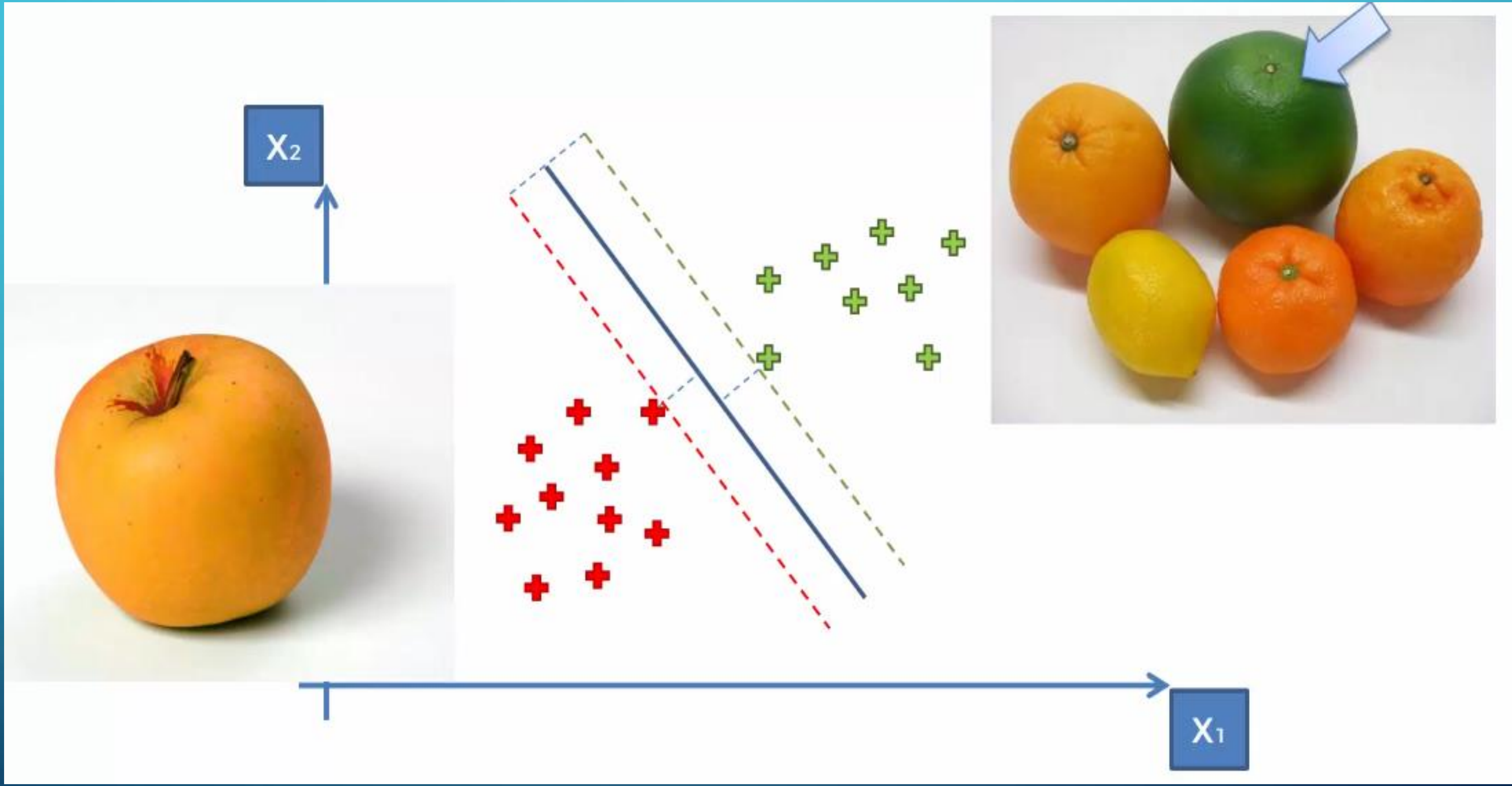


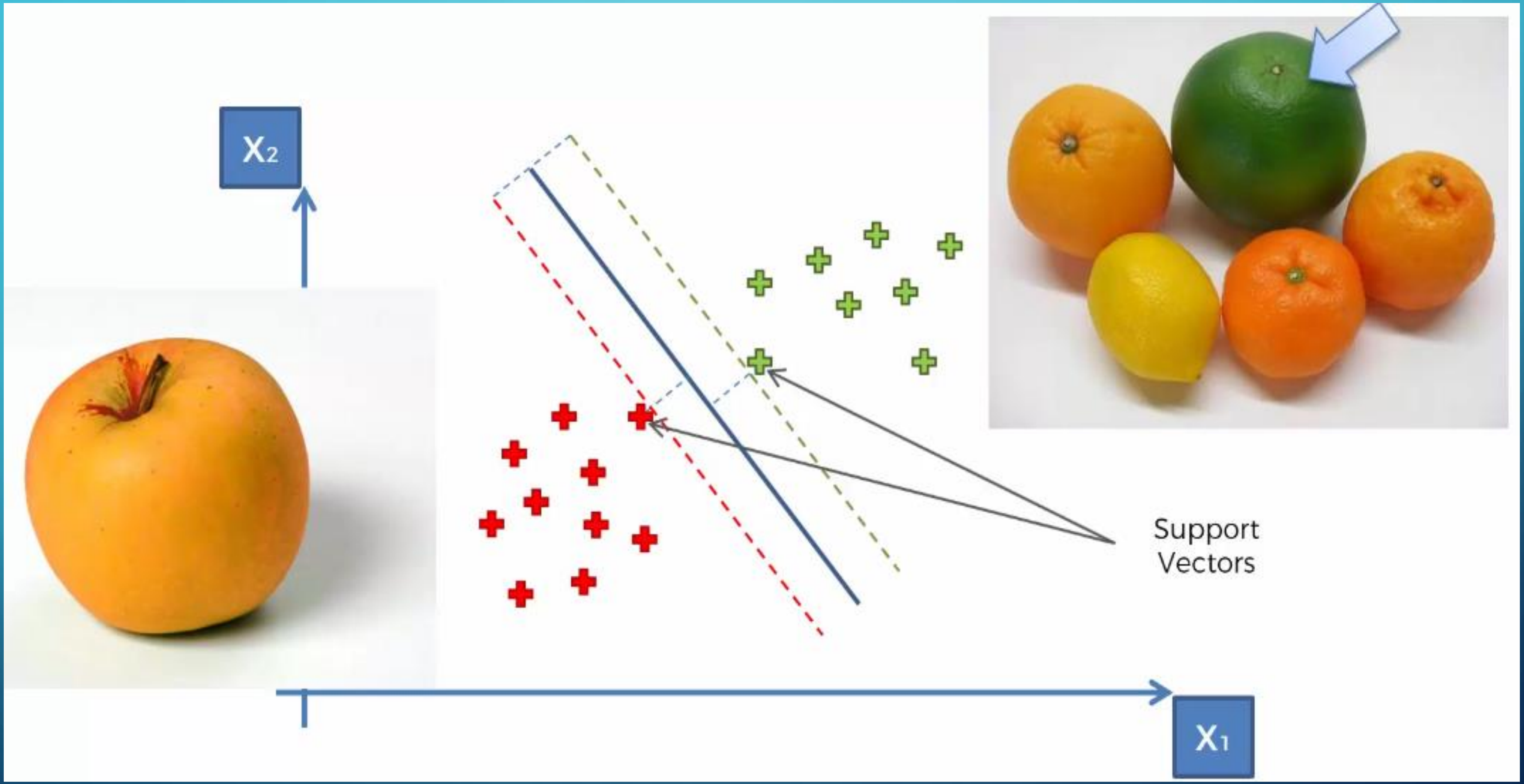
SUPPORT VECTORS



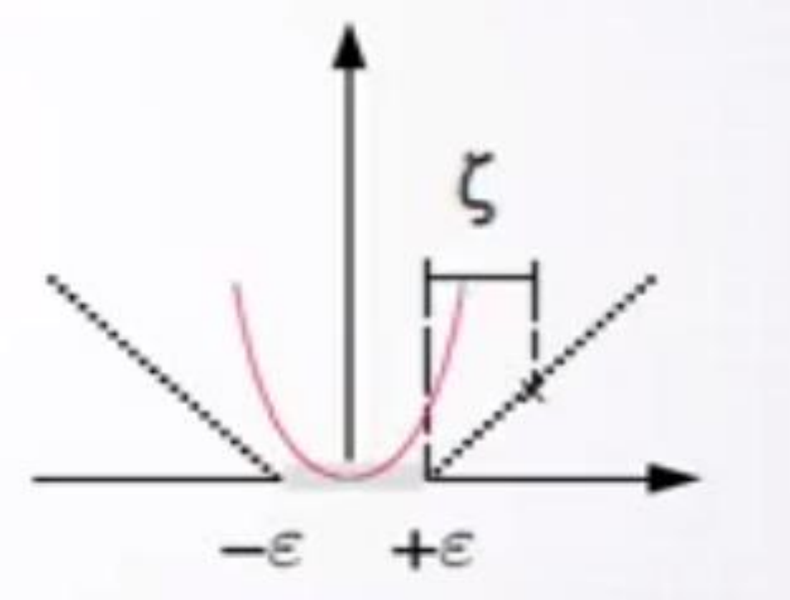
WHAT IS SO SPECIAL??


X2
X1


X2
X1

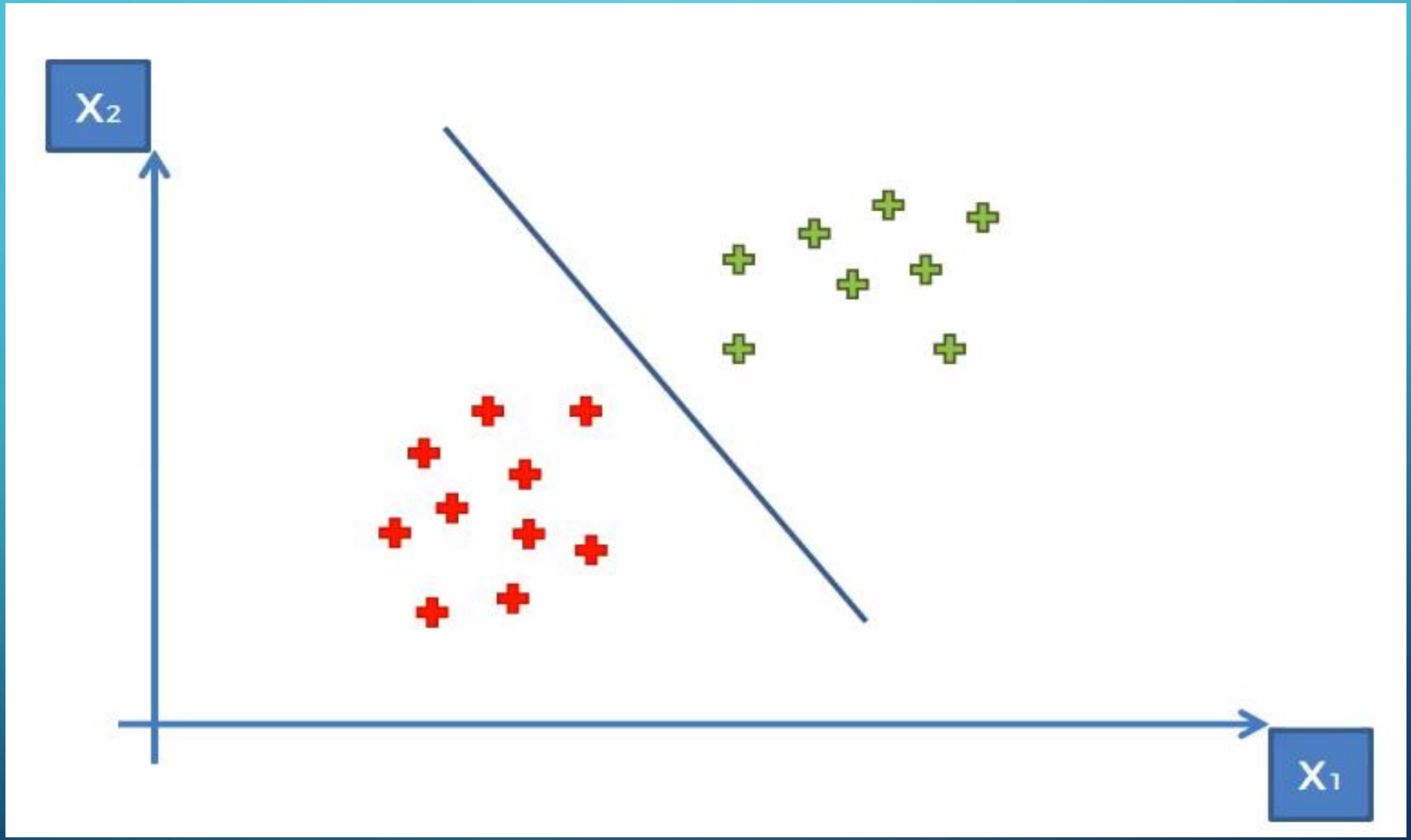

X2
Support
Vectors
X1

COST FUNCTION INTERPRETATION

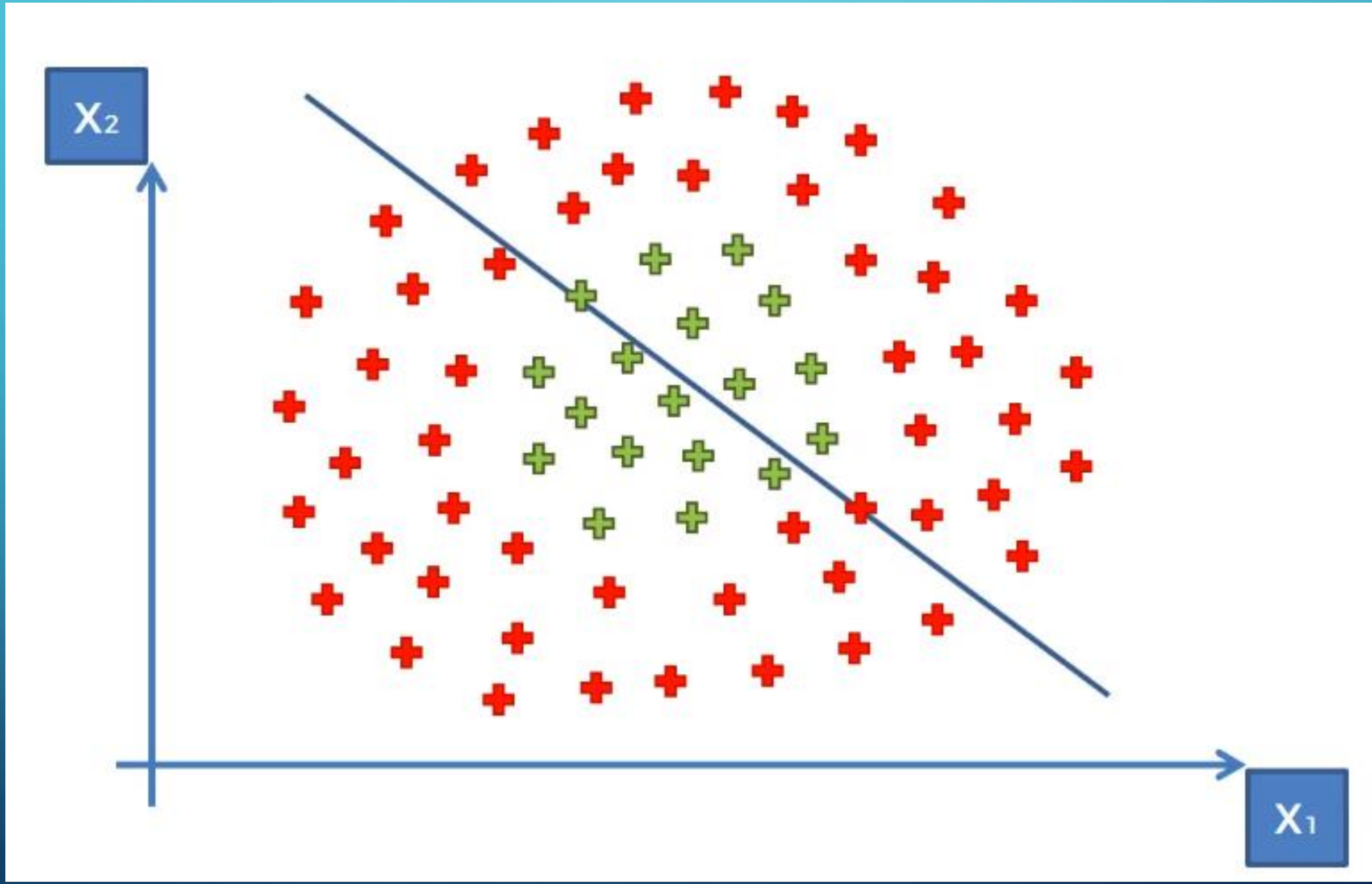


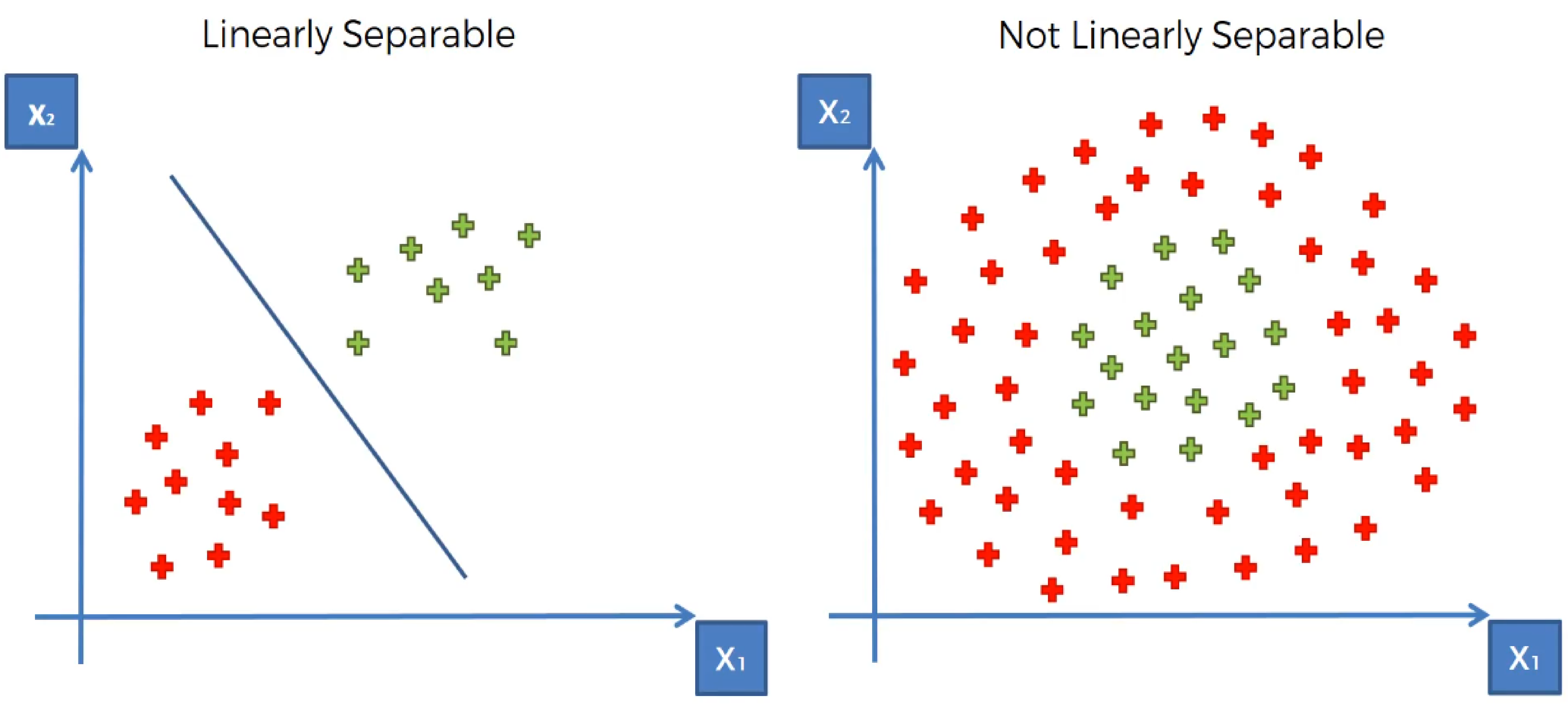
KERNEL SVM INTUITION

SVM WORKED WELL FOR THIS!!



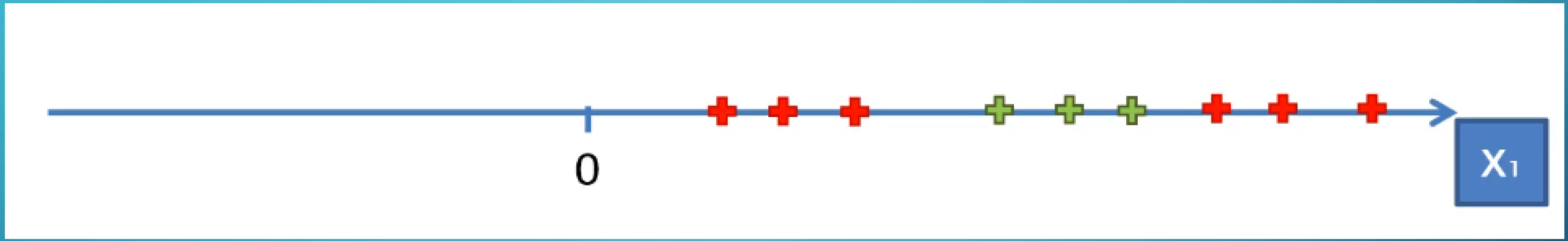
VANILLA SVM CAN ONLY DRAW LINEAR BOUNDARY!!

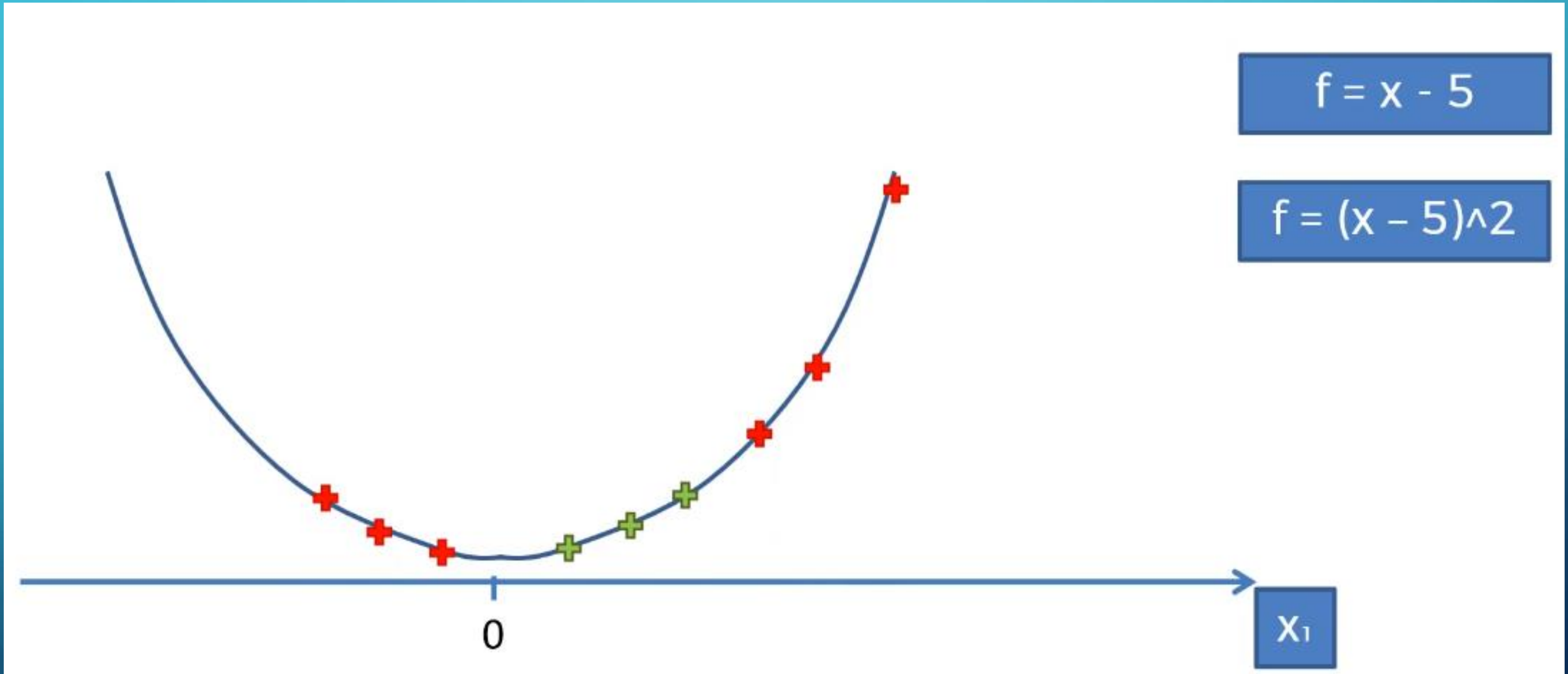



Linearly Separable
Not Linearly Separable
X2
X1
X2
X1

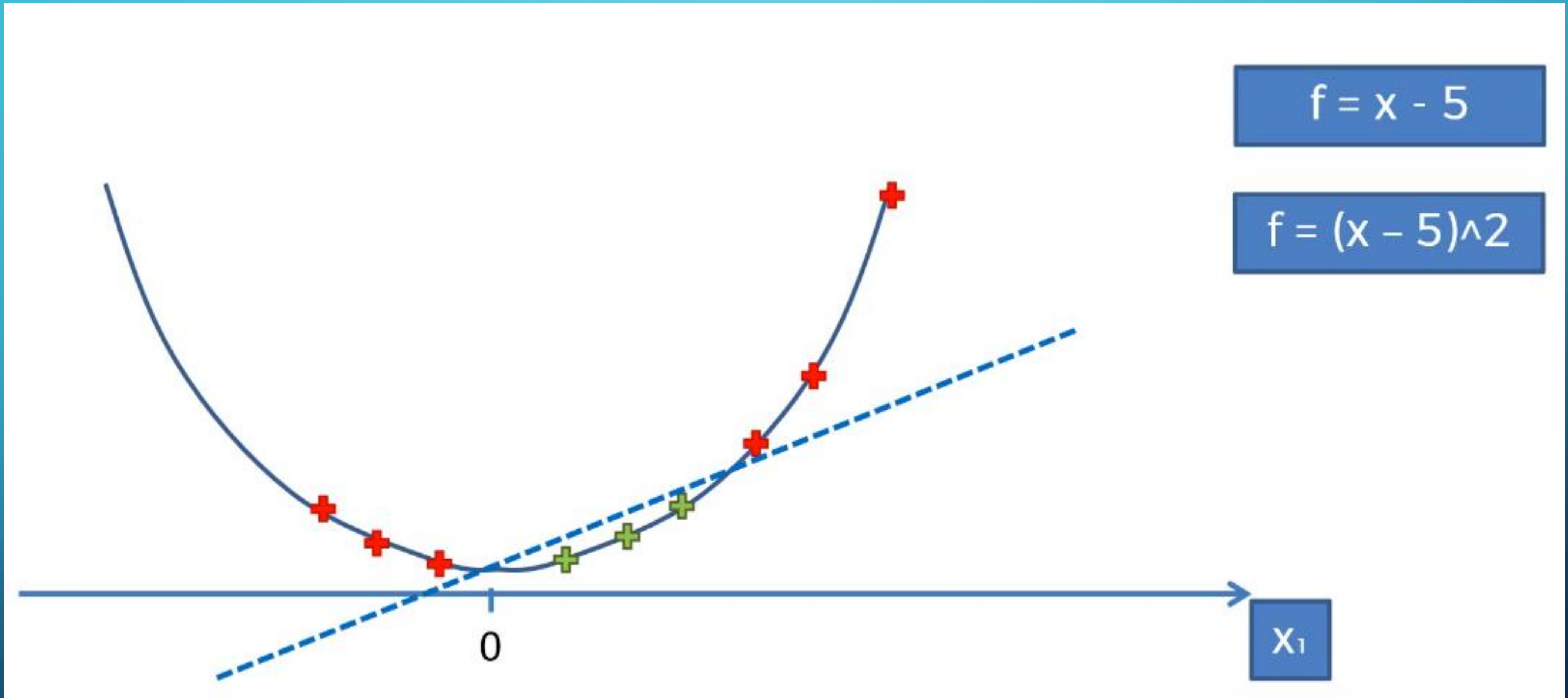
HIGHER DIMENSIONS TO THE RESCUE!!

VANILLA SVM CANT CLASSIFY THIS!!

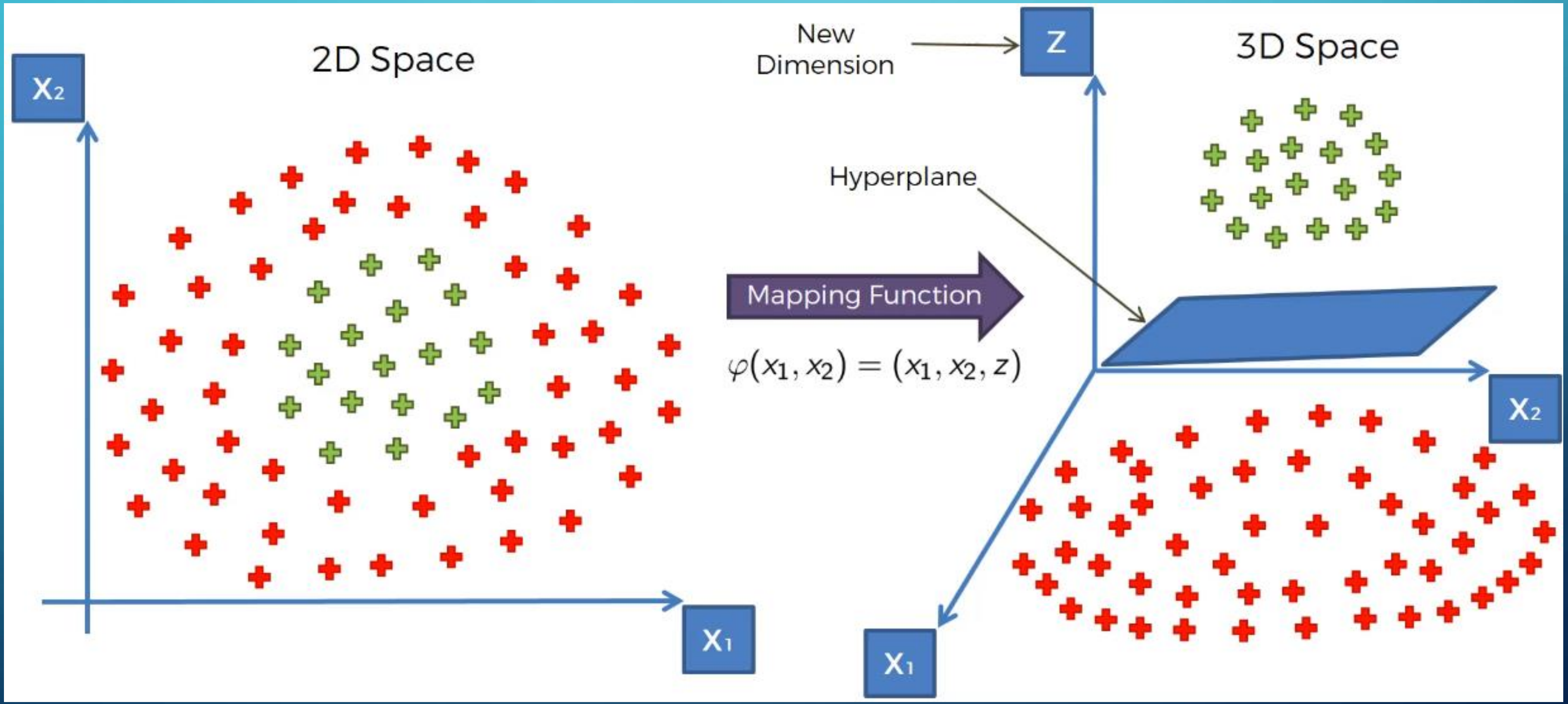




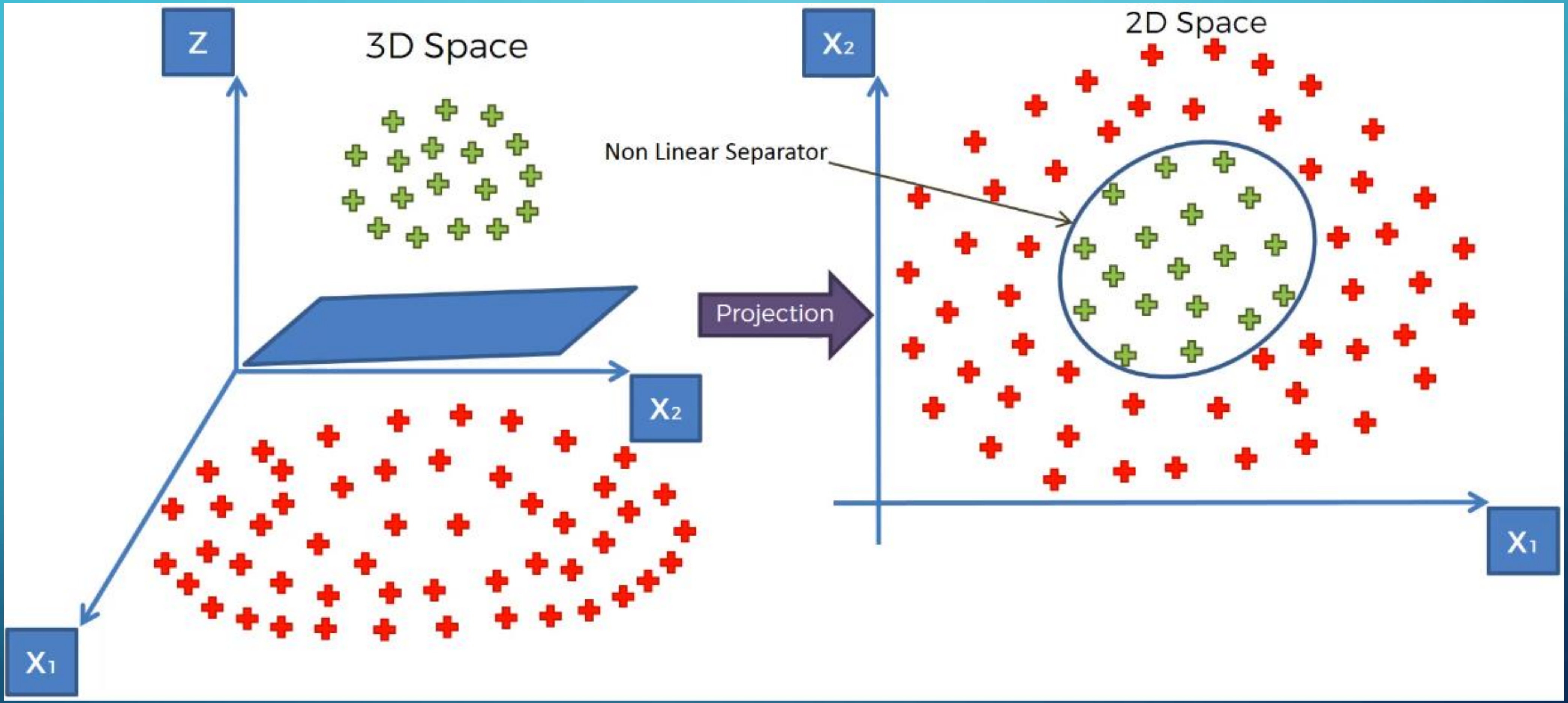
f = x - 5
f = (x – 5)^2
0
x1


f = x - 5
f = (x – 5)^2
0
x1

SIMILARLY!!



2D Space
X2
X1
New Dimension
Z
3D Space
Hyperplane
Mapping Function
$\varphi(x_1, x_2) = (x_1, x_2, z)$
X2
X1



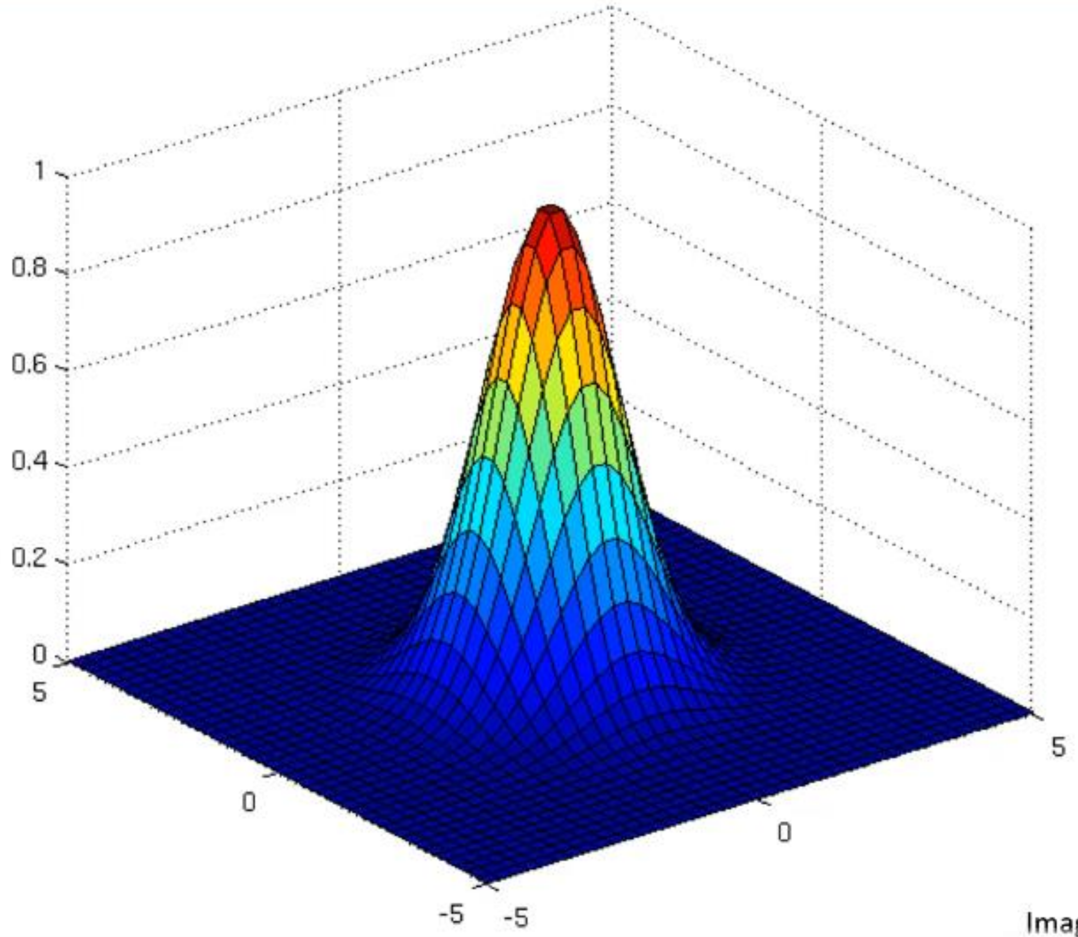
Z
3D Space
X2
X1
Projection
X2
2D Space
Non Linear Separator
X1

Mapping to higher dimension can be computationally expensive!!

KERNEL TRICK

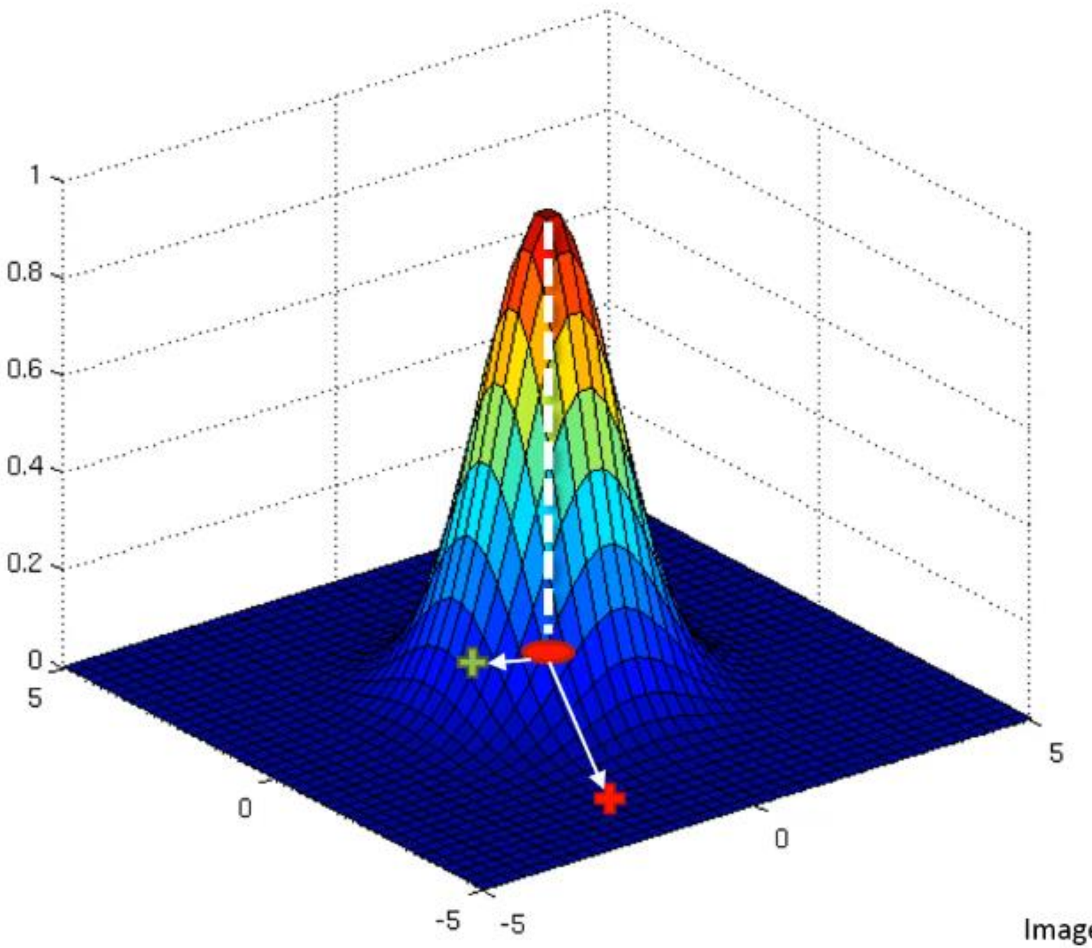
GAUSSIAN RBF KERNEL

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



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

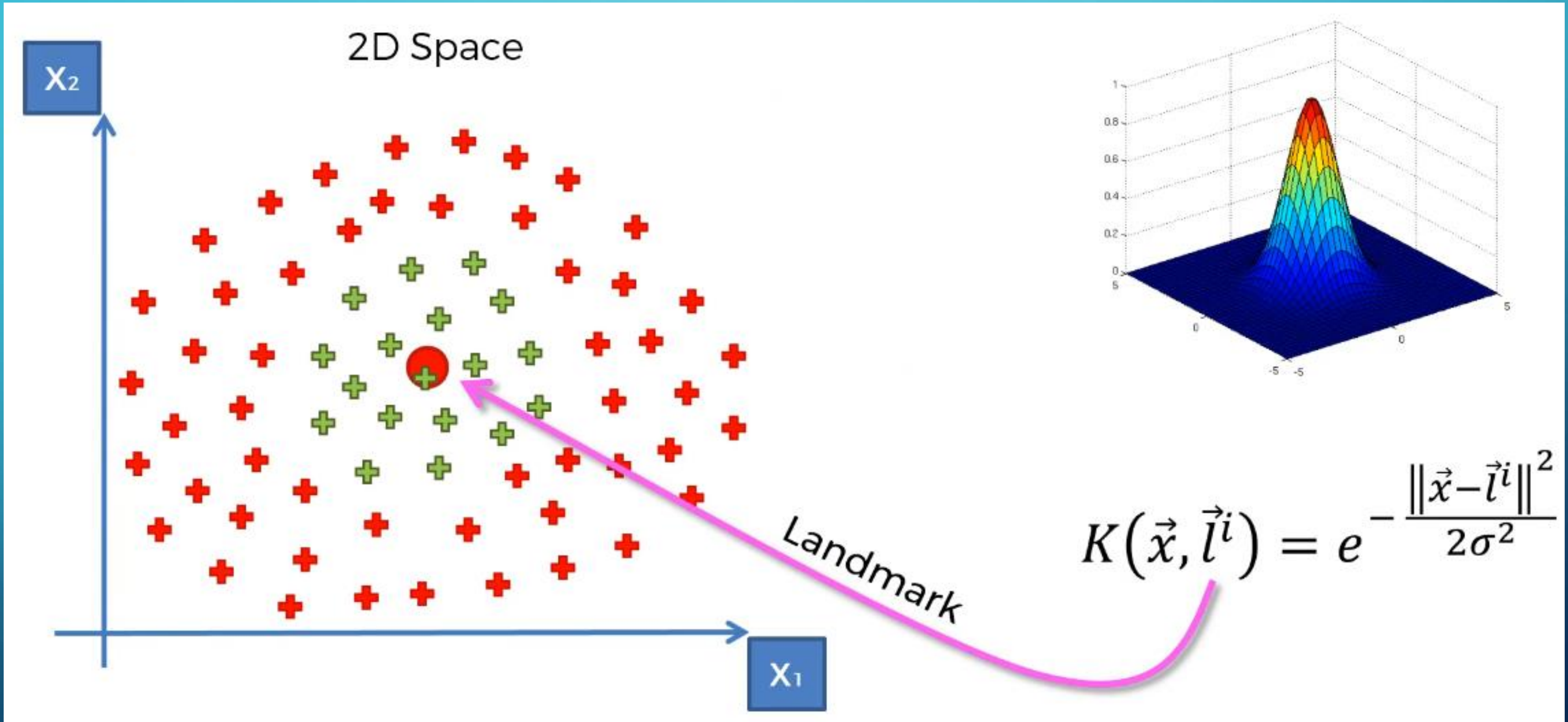
Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

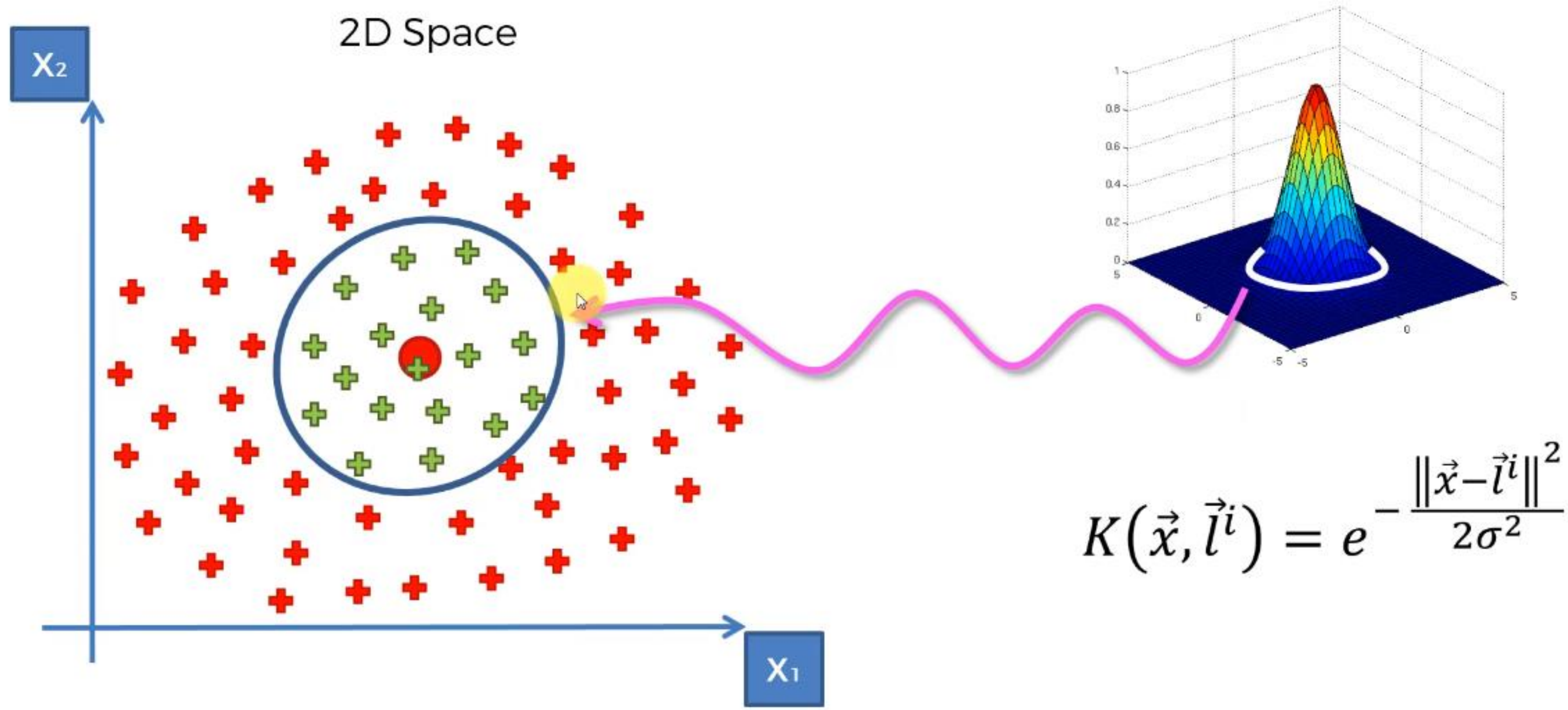


$$K(\vec{x}, \vec{l}^{i}) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

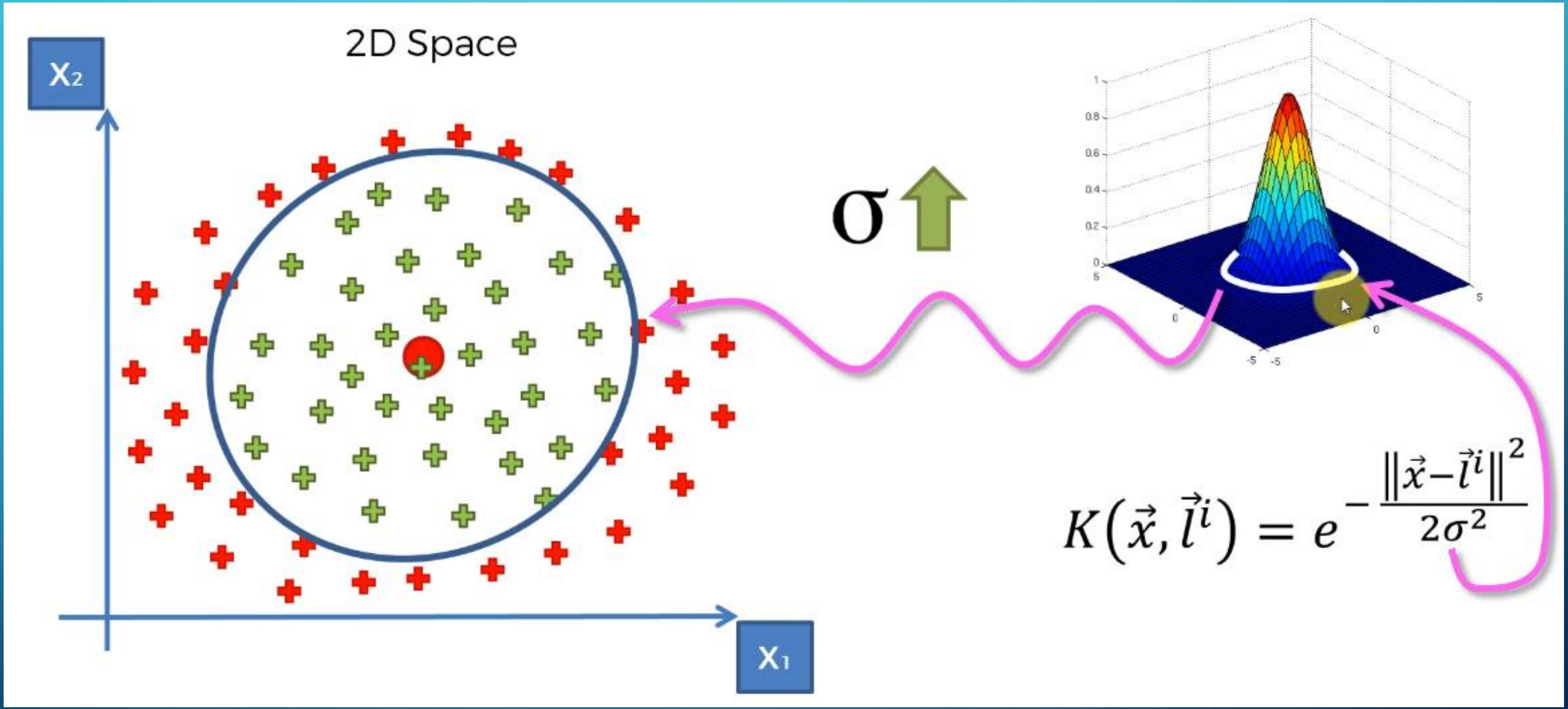
Image source: http://www.cs.toronto.edu/~duvenaud/cookbook/index.html

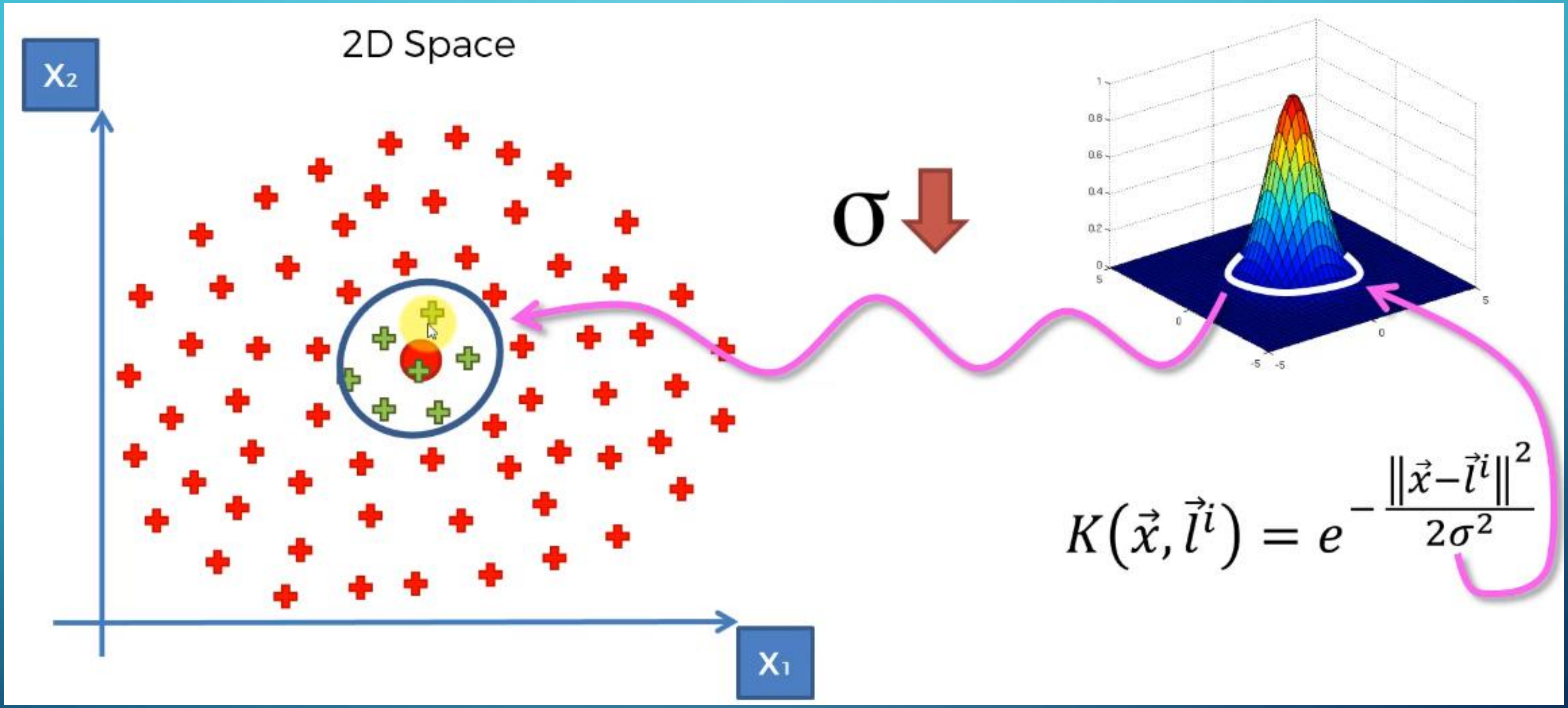
HOW IT WORKS!!

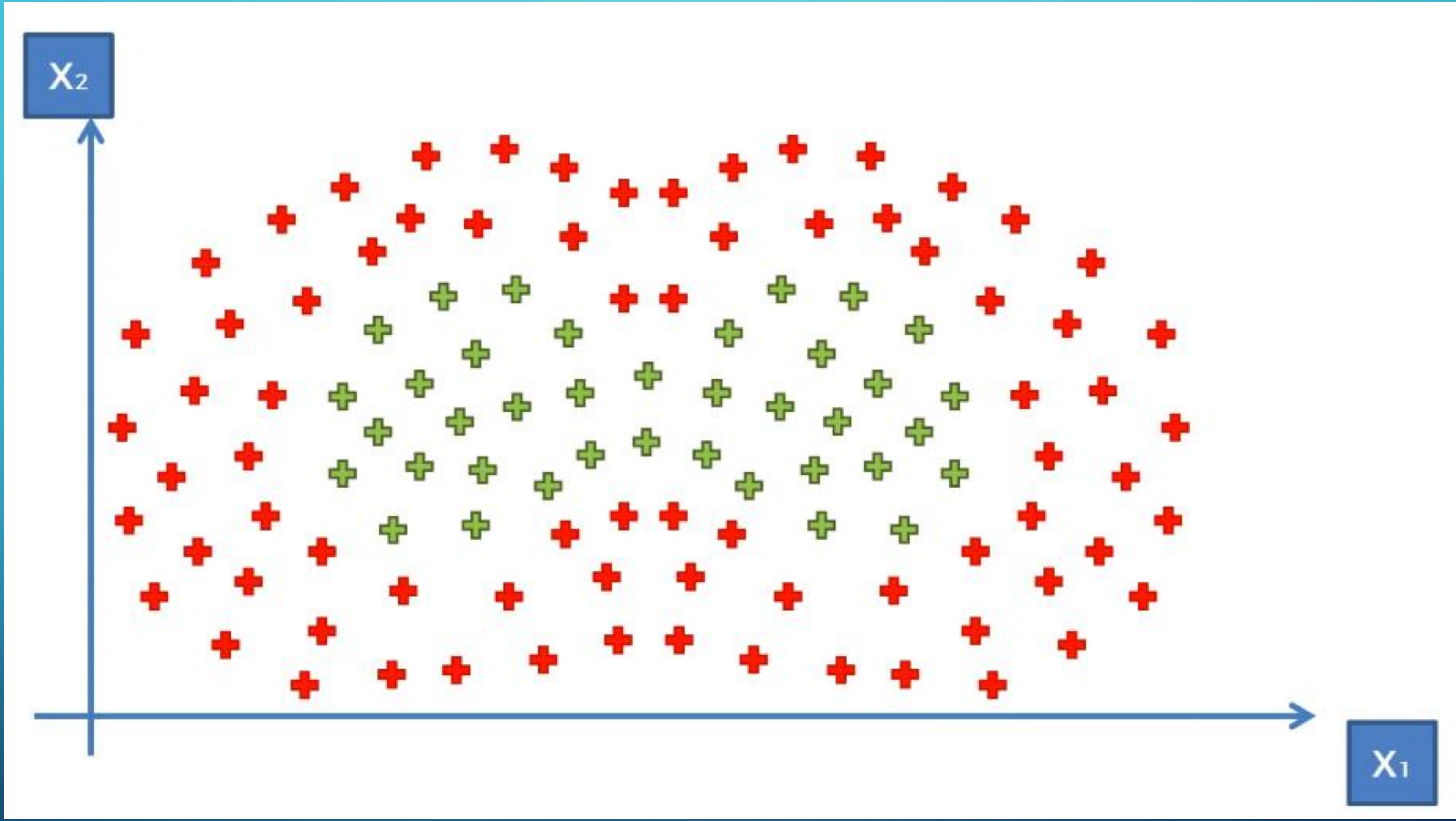

2D Space
X_2
X_1
Landmark
$K(\vec{x},\vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$

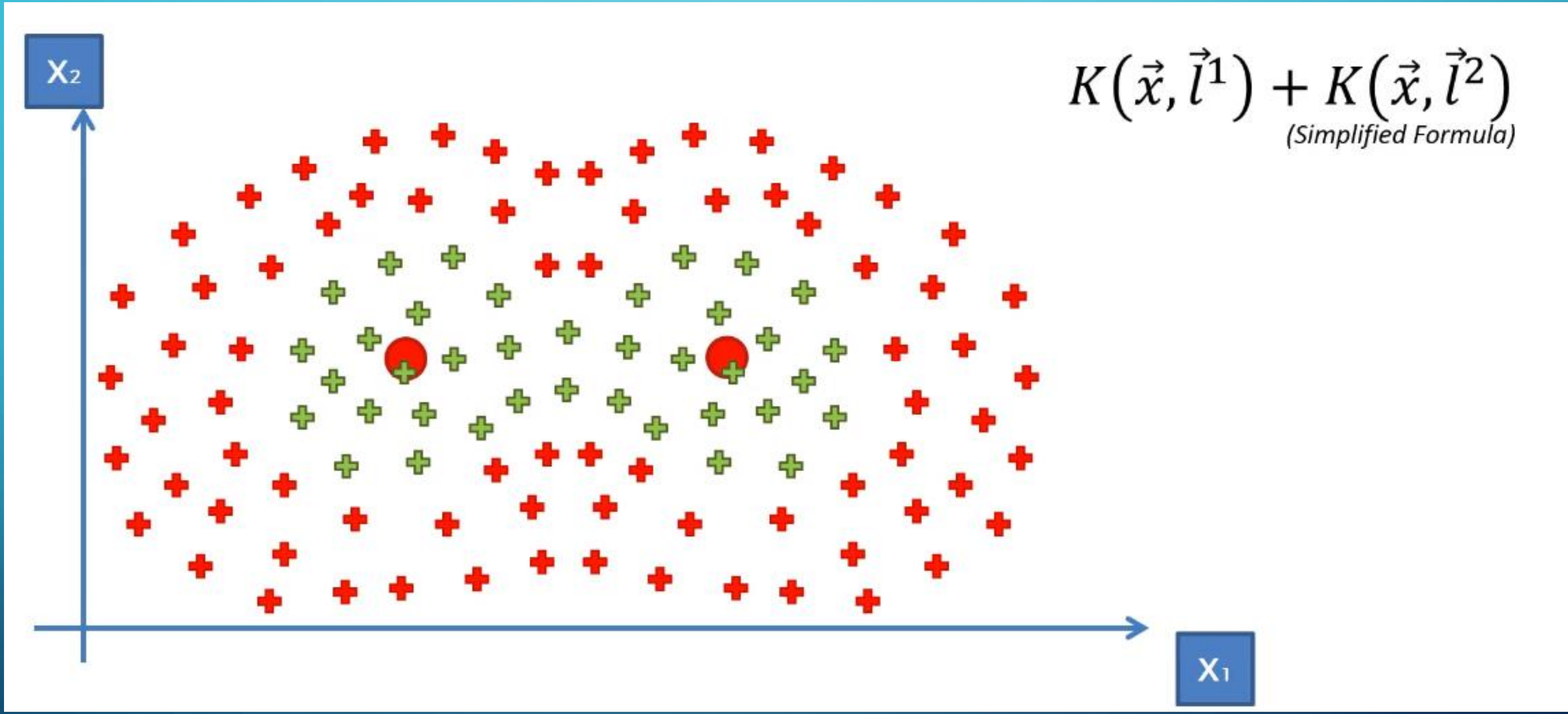


$$K(\vec{x}, \vec{l}^{i}) = e^{-\frac{\|\vec{x}-\vec{l}^{i}\|^2}{2\sigma^2}}$$

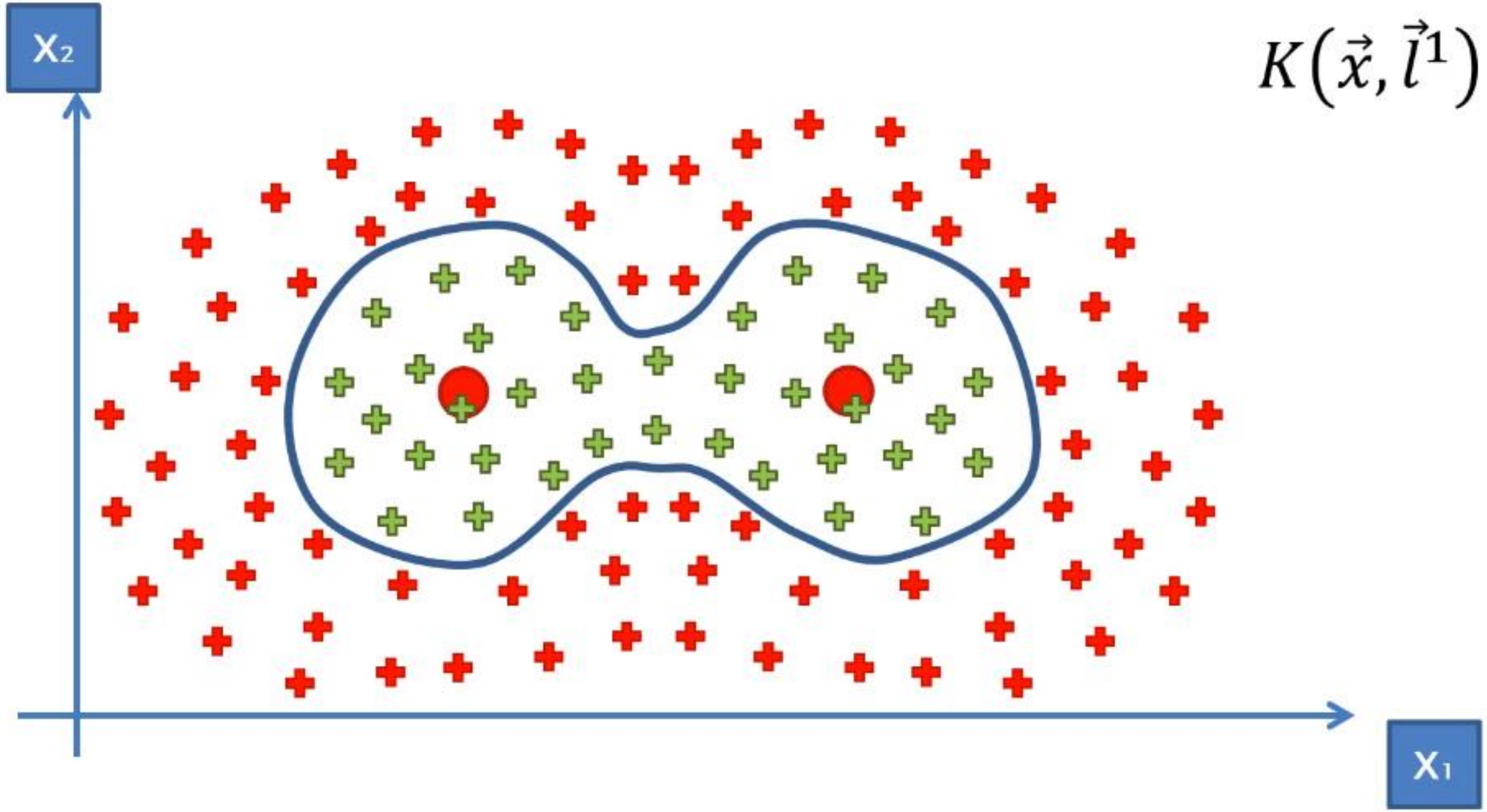

2D Space
X_2
X_1
σ
$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$


2D Space
X2
X1
σ
$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x}-\vec{l}^i\|^2}{2\sigma^2}}$


X2
X1

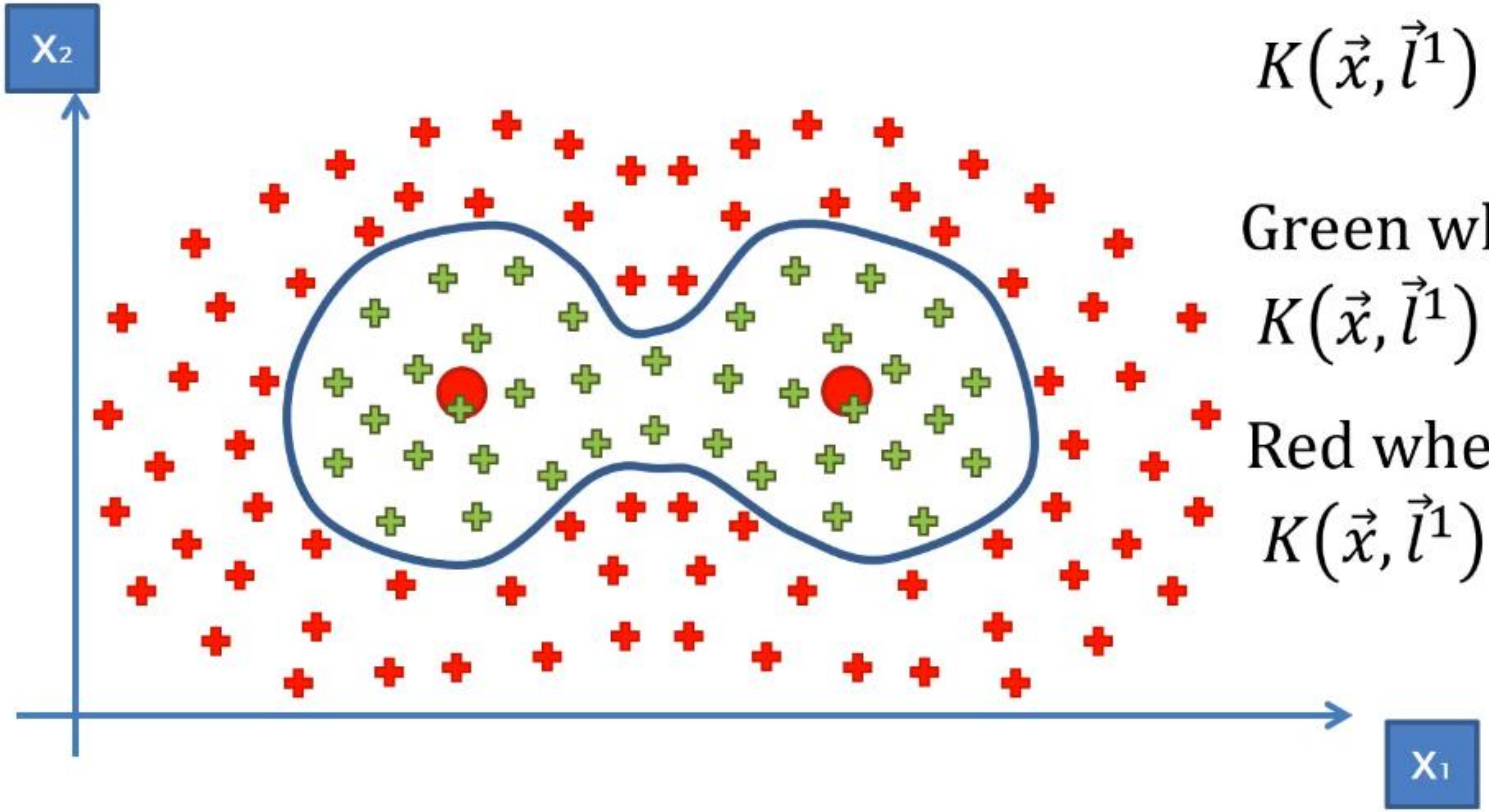


X_2
$K(\vec{x}, \vec{l}^1) + K(\vec{x}, \vec{l}^2)$
(Simplified Formula)
X_1



$$K\left(\vec{x}, \vec{l}^{1}\right) + K\left(\vec{x}, \vec{l}^{2}\right)$$

(Simplified Formula)



$$K(\vec{x},\vec{l}^1) + K(\vec{x},\vec{l}^2)$$

(Simplified Formula)

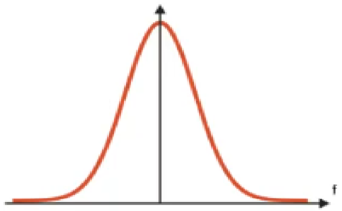
Green when:

$$K(\vec{x},\vec{l}^1) + K(\vec{x},\vec{l}^2) > 0$$

Red when:

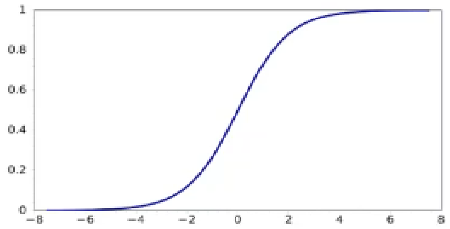
$$K(\vec{x},\vec{l}^1) + K(\vec{x},\vec{l}^2) = 0$$

TYPES OF KERNEL FUNCTIONS



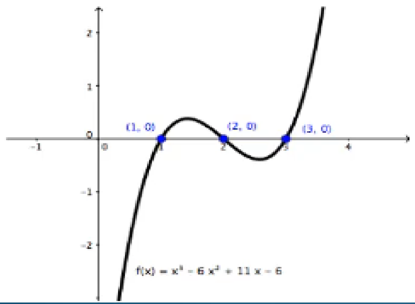
Gaussian RBF Kernel

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



Sigmoid Kernel

$$K(X,Y) = \tanh(\gamma \cdot X^{T}Y + r)$$



Polynomial Kernel

$$K(X,Y) = (\gamma \cdot X^{T}Y + r)^{d}, \gamma > 0$$

ADVANTAGES:

- Works well with even unstructured and semi structured data like text, Images.
- The kernel trick is real strength of SVM. With an appropriate kernel function, we can solve any complex problem. Unlike in neural networks, SVM is not solved for local optima
- It scales relatively well to high dimensional data
- SVM models have generalization in practice, the risk of over-fitting is less in SVM
- SVM is always compared with ANN. When compared to ANN models, SVMs give better results

DISADVANTAGES

- Choosing a “good” kernel function is not easy
- Long training time for large datasets
- Difficult to understand and interpret the final model, variable weights and individual impact
- Since the final model is not so easy to see, we can not do small calibrations to the model hence its tough to incorporate our business logic
- The SVM hyper parameters are Cost -C and gamma. It is not that easy to fine-tune these hyper-parameters. It is hard to visualize their impact

APPLICATIONS

- Protein Structure Prediction
- Handwriting Recognition
- Detecting Steganography in digital images
- Breast Cancer Diagnosis
- Almost all the applications where ANN is used

CONCLUSION

- Many software tools are available for SVM implementation
- SVMs are really good for text classification
- SVMs are good at finding the best linear separator. The kernel trick makes SVMs non-linear learning algorithms
- Choosing an appropriate kernel is the key for good SVM and choosing the right kernel function is not easy
- We need to be patient while building SVMs on large datasets. They take a lot of time for training

NEXT WEEK

- Problem of Bias and Overfitting
- Using graphs to resolve the issues in ML
- Concept of Precision and Recall
- Writing codes for all classification algorithms

THANK YOU