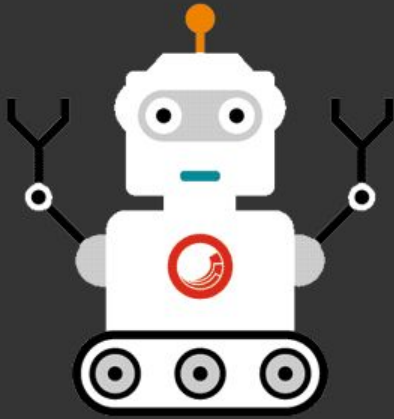


# Support Vector Machine

# How to train your machine





**ARTIFICIAL**  
Intelligence



**Machine**  
learning



**DEEP**  
learning

# HOW TO TRAIN YOUR MACHINE

AI enables the machine to think and take its own decision

The diagram features a large, light gray rectangular background. On the left side, there is a vertical white bar. Three speech bubble-like shapes are positioned around this bar. The top bubble is dark teal and points to the right. The bottom bubble is a medium teal and points to the right. The rightmost bubble is a light blue-gray and points to the left. All three bubbles have a tail pointing towards the central gray area.

Deep Learning is basically mimicking the human brain

uses statistical learning algorithms to build systems that can automatically learn and improve from experiences

What is Machine Learning?



## supervised learning

Input data



Annotations

These are  
apples



Model

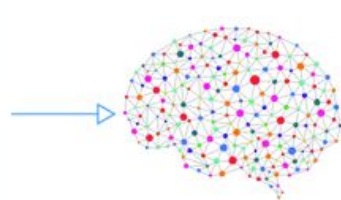


Prediction

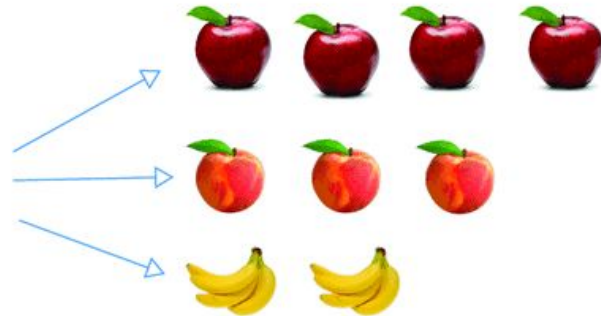
Its an  
apple!

## unsupervised learning

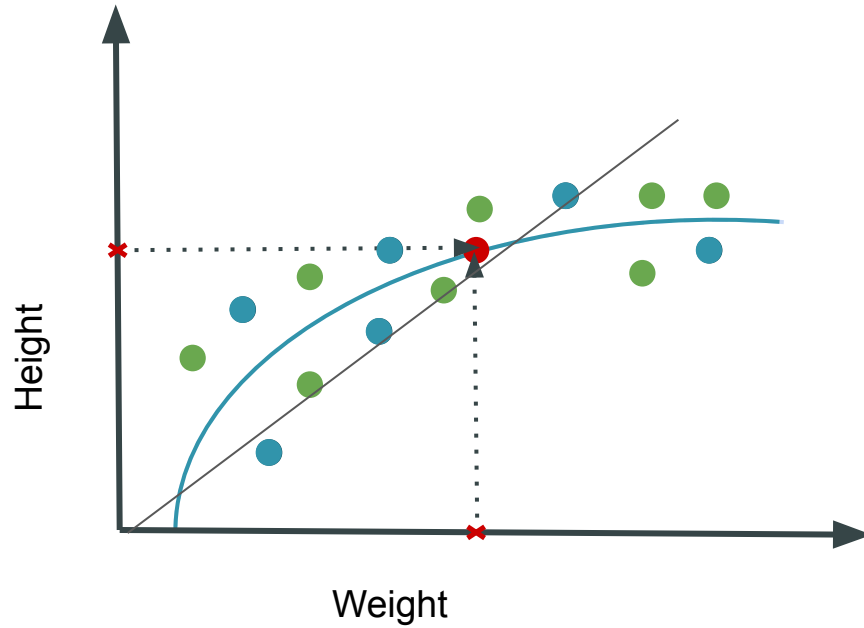
Input data



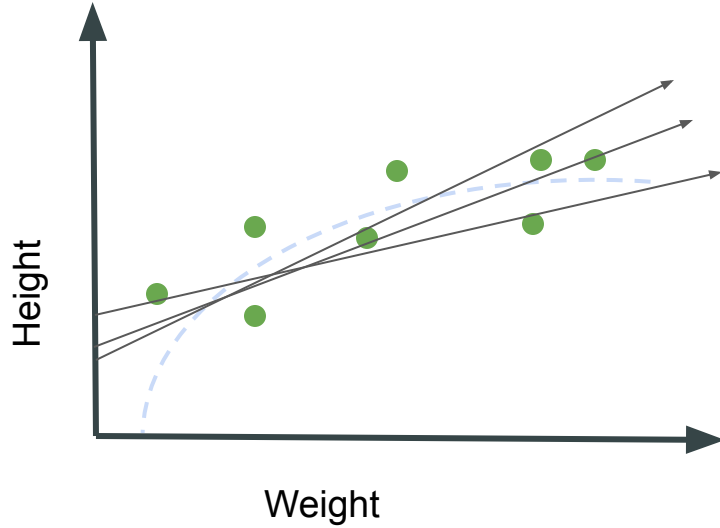
Model



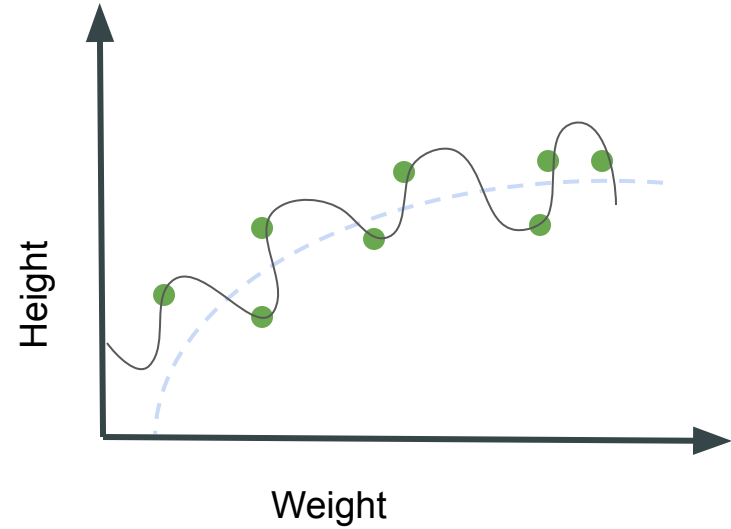
# Training Data and Test Data



# Bias

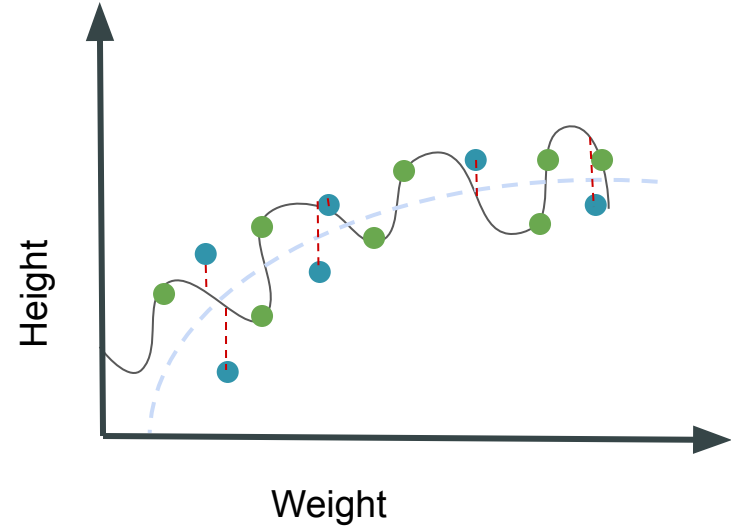
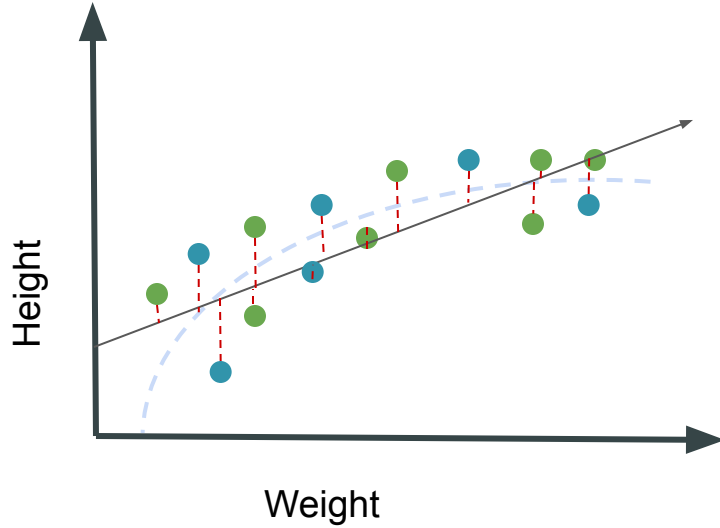


The **inability** for a machine learning method **to capture the true relationship** is called **bias**



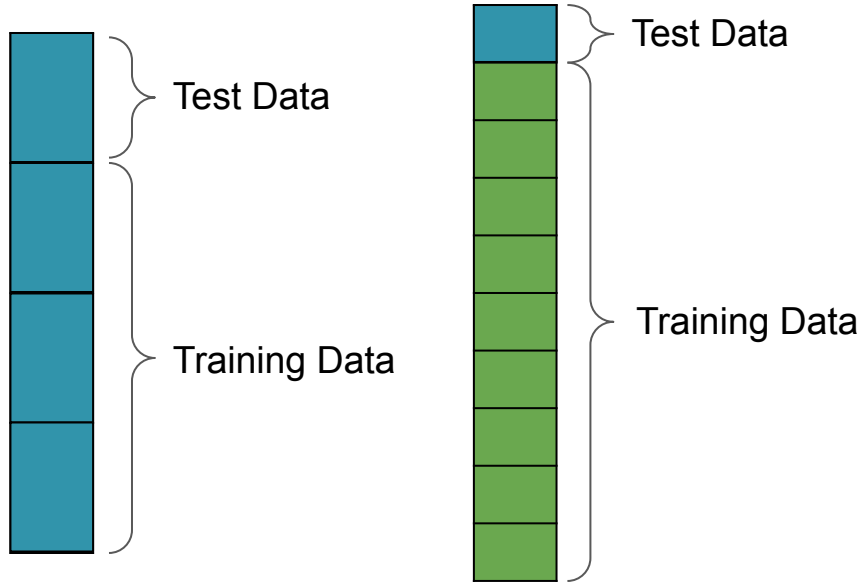


# Variance



The difference in fits  
between data sets is called  
**variance**

# CROSS VALIDATION



- Test Data
- Training Data

How do you decide which portion of your dataset is test data and which portion is training data??

Divide whole dataset in  $k$  portions

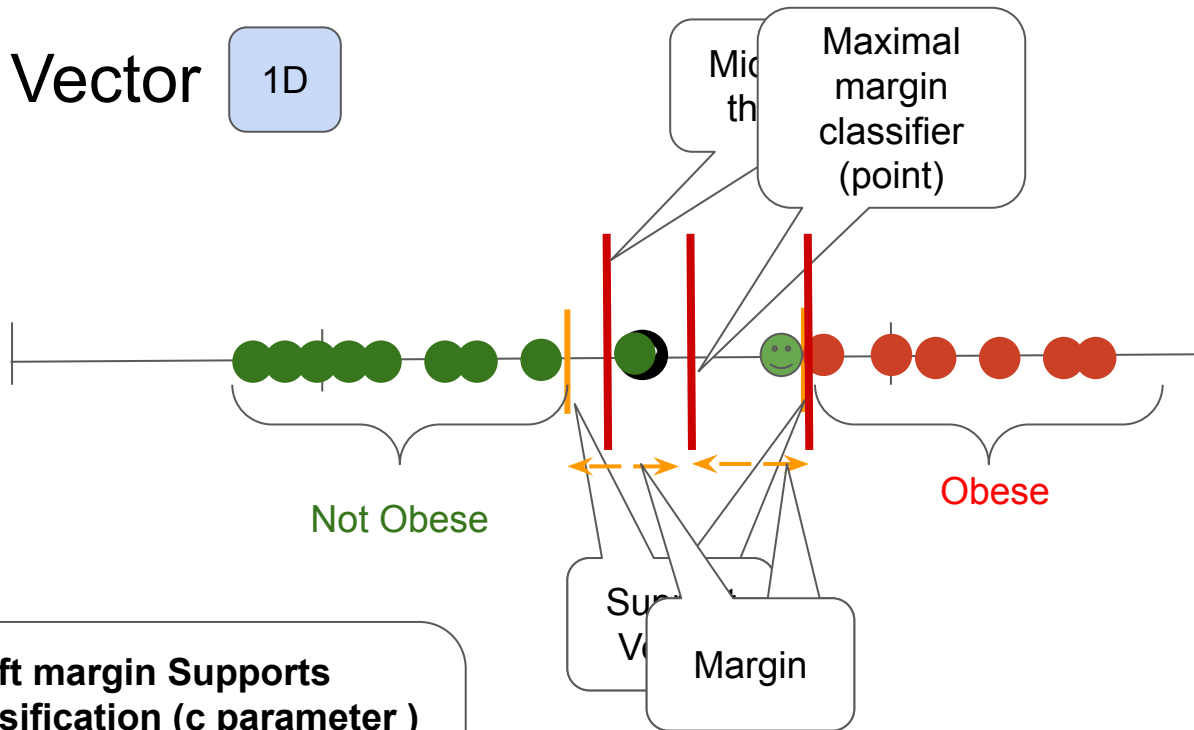
Make one portion test data each time and use rest to train the model

Summarize the results at end

# Support Vector

1D

Mass(g):



Not Obese

Obese

**Soft margin Supports  
misclassification (c parameter )**

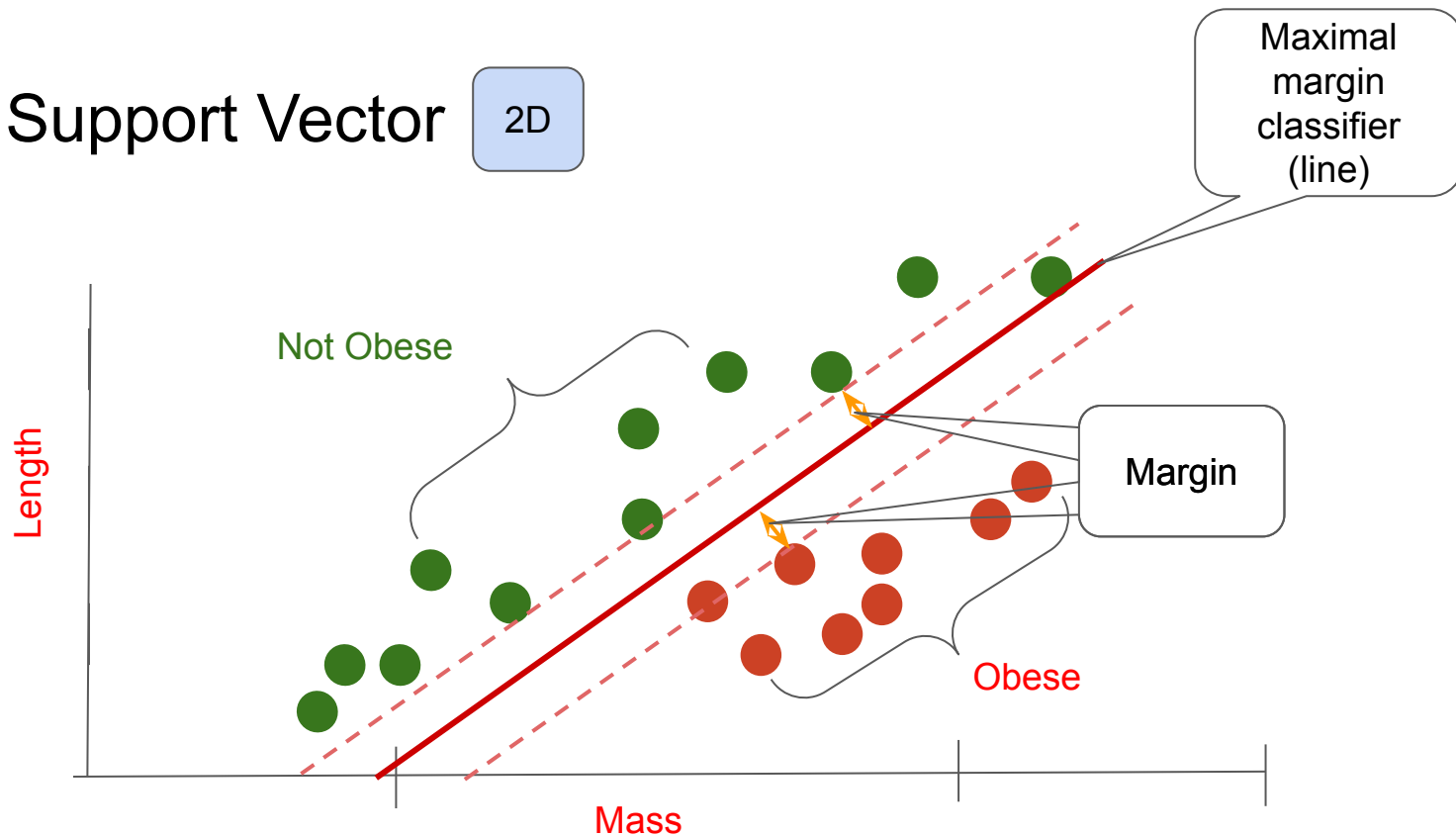
**Higher bias**

**Lower variance**

**Low c**

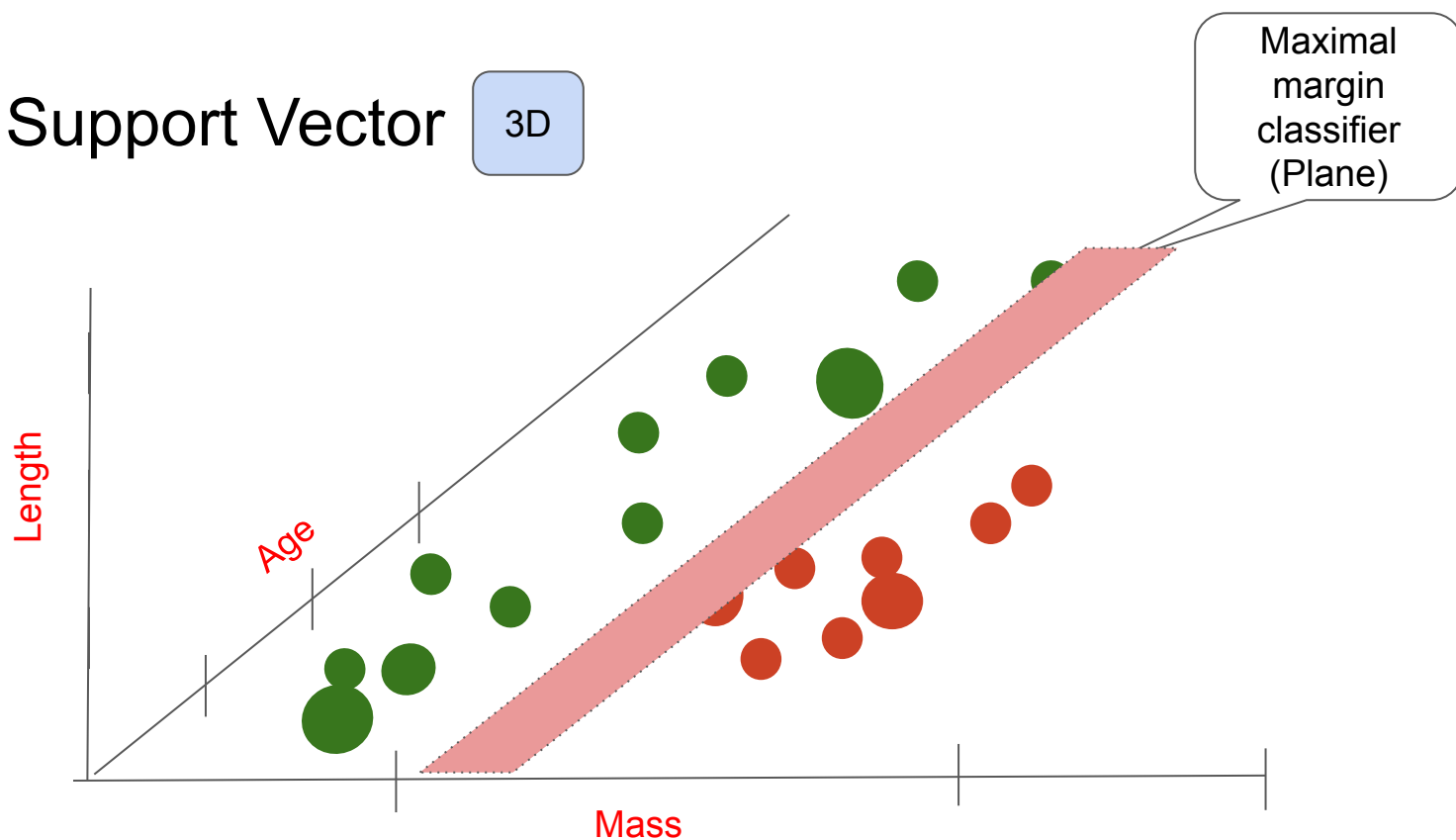
# Support Vector

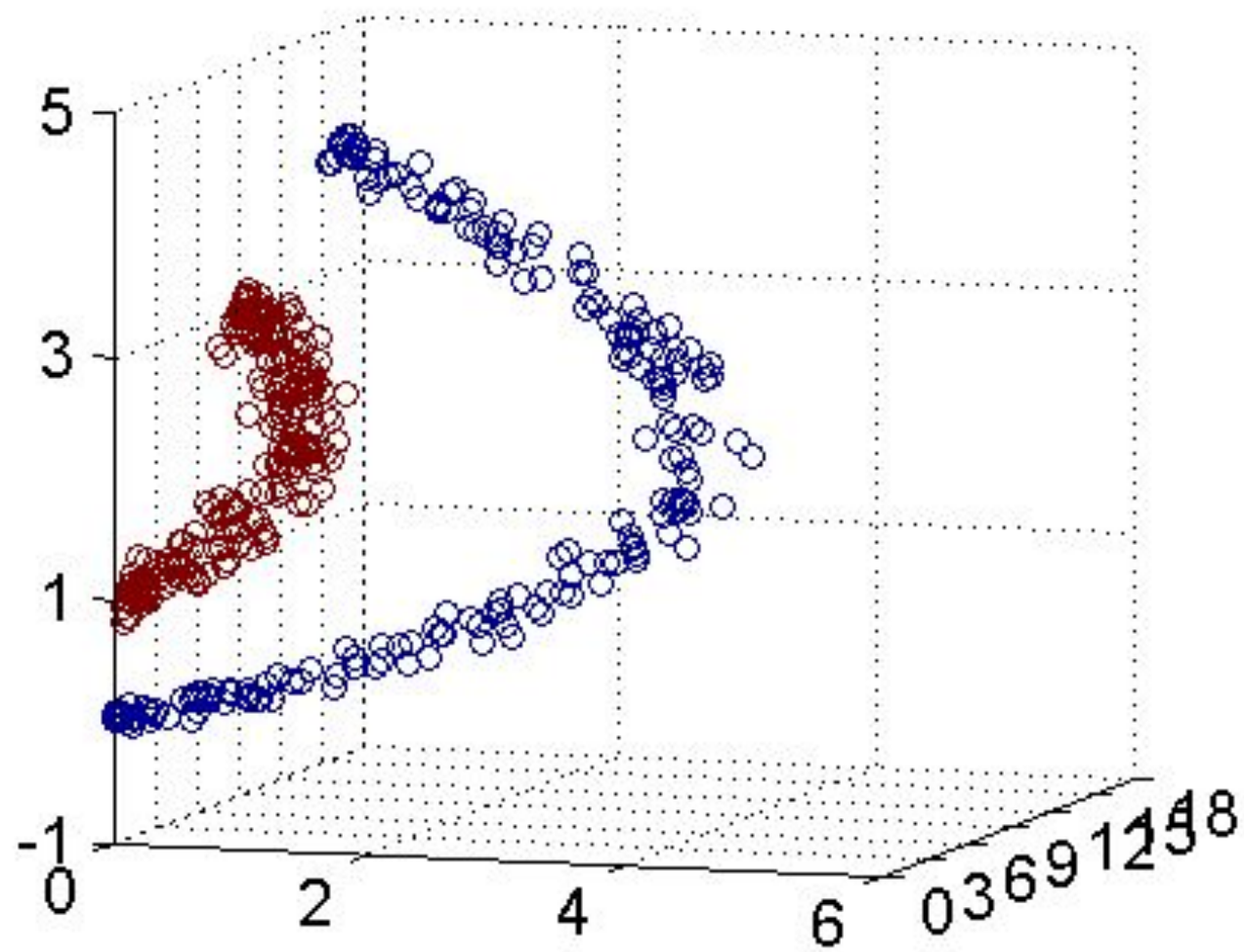
2D



# Support Vector

3D





Did you notice  
something  
weird?

Support vector  
Classifier in a 1D  
line is a **single point**

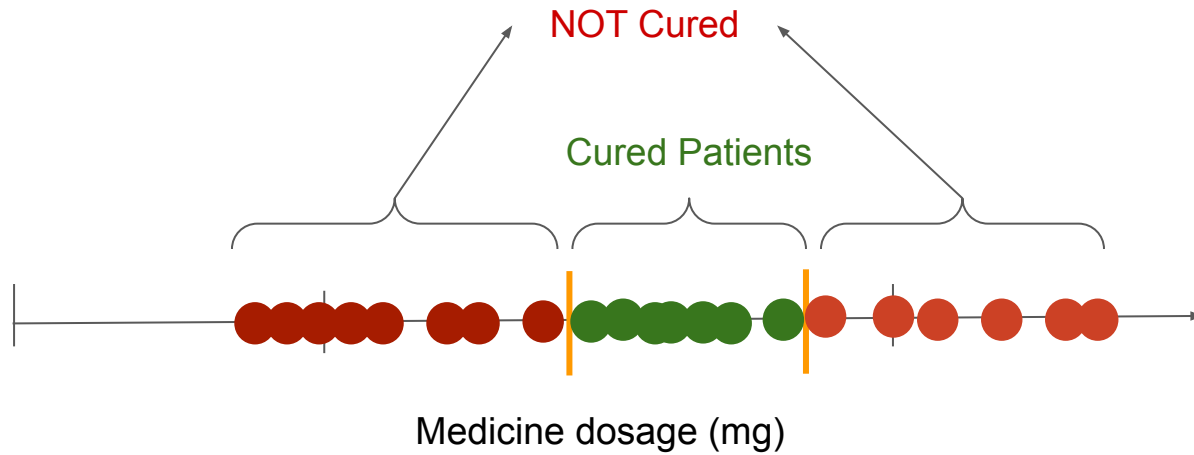
Support vector  
Classifier in a  
3D space is a  
**2D plane**

Support vector  
Classifier in a 2D  
space is a **1D line**

Support vector  
Classifier in a 4 or  
more dimensional data  
space is a  
**HYPERPLANE**

# Support Vector Machine

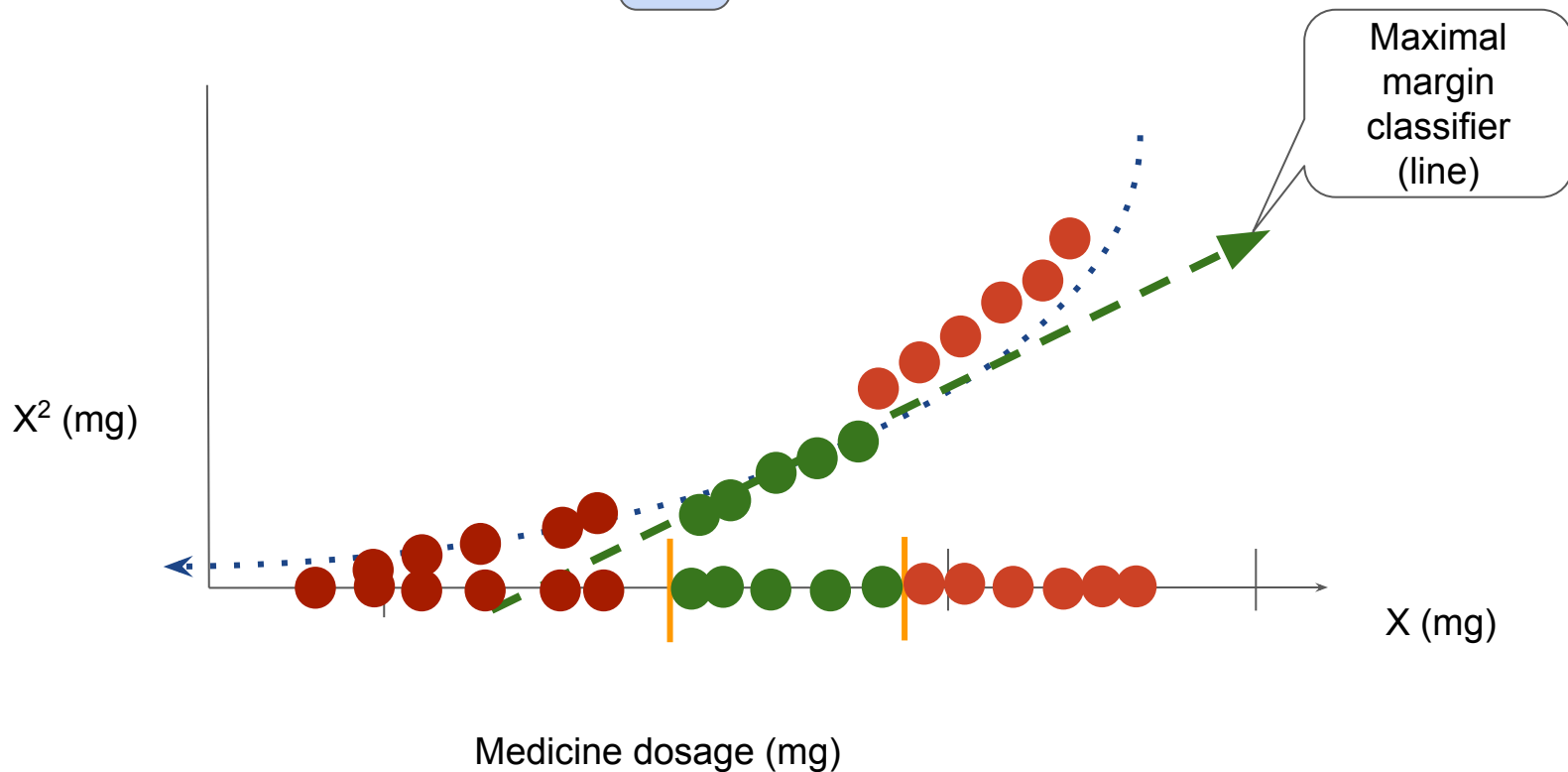
1D





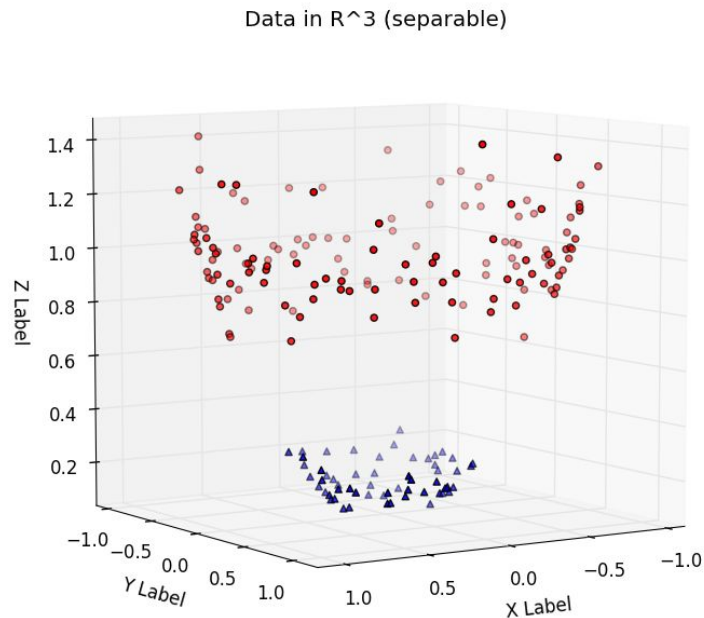
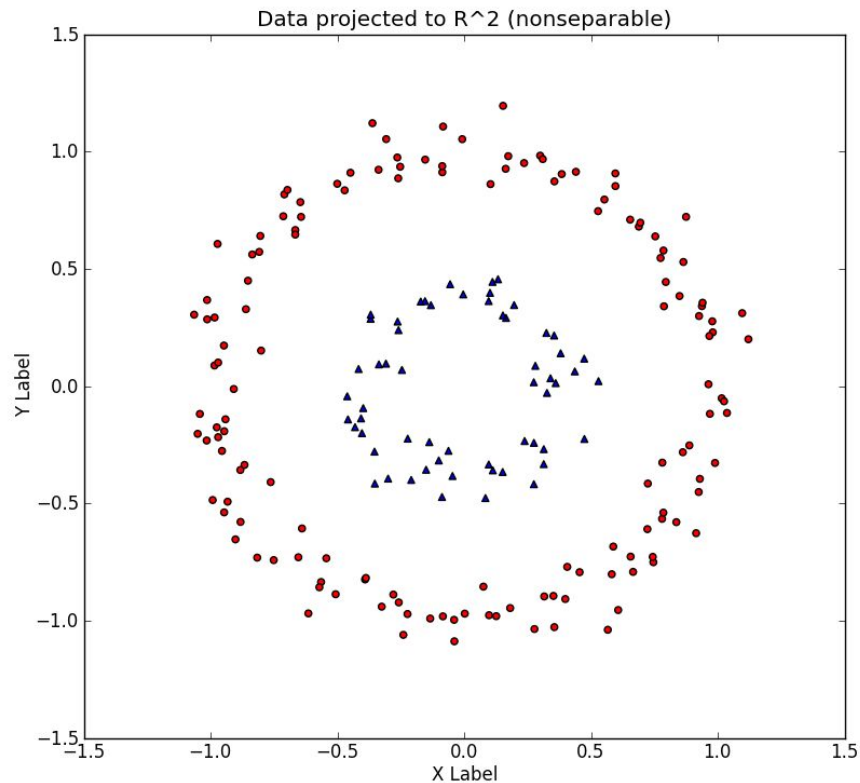
# Support Vector Machine<sub>(cont)</sub>

1D



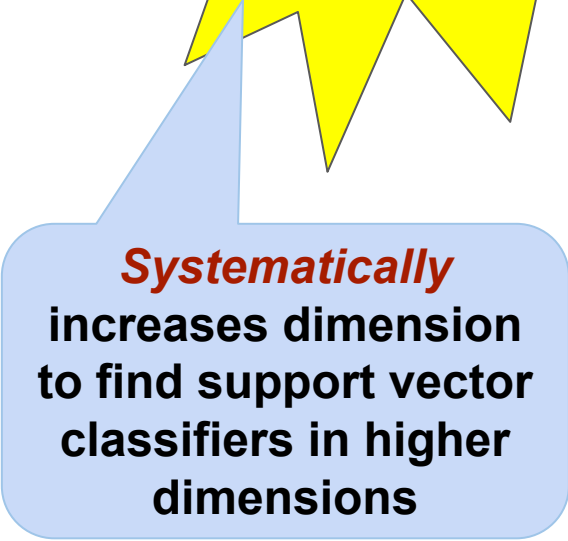
# Support Vector Machine<sub>(cont)</sub>

2D






## THE KERNEL FUNCTION

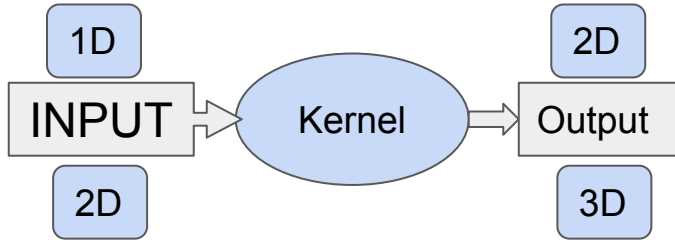


***Systematically***  
increases dimension  
to find support vector  
classifiers in higher  
dimensions



**Why  $x^2$ ?!**  
How do we decide  
how to **transform**  
the data?

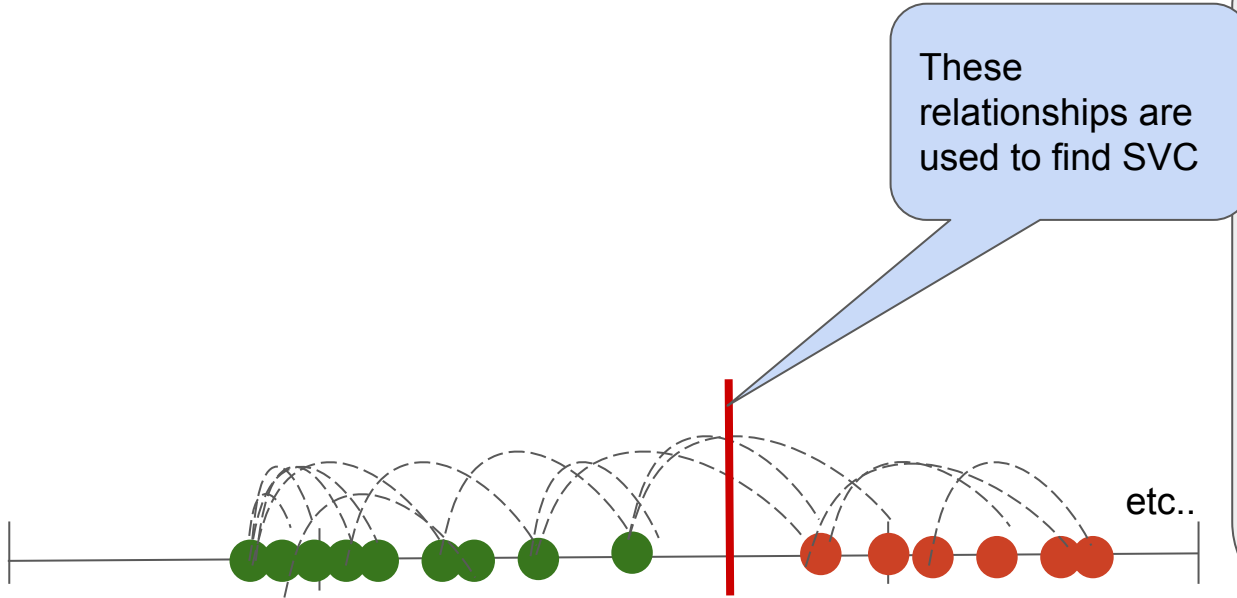
# KERNEL Function



Different types of Kernel functions:

- Linear
- Nonlinear
- Polynomial
- Radial basis function (RBF)
- Sigmoid

# Polynomial Kernel

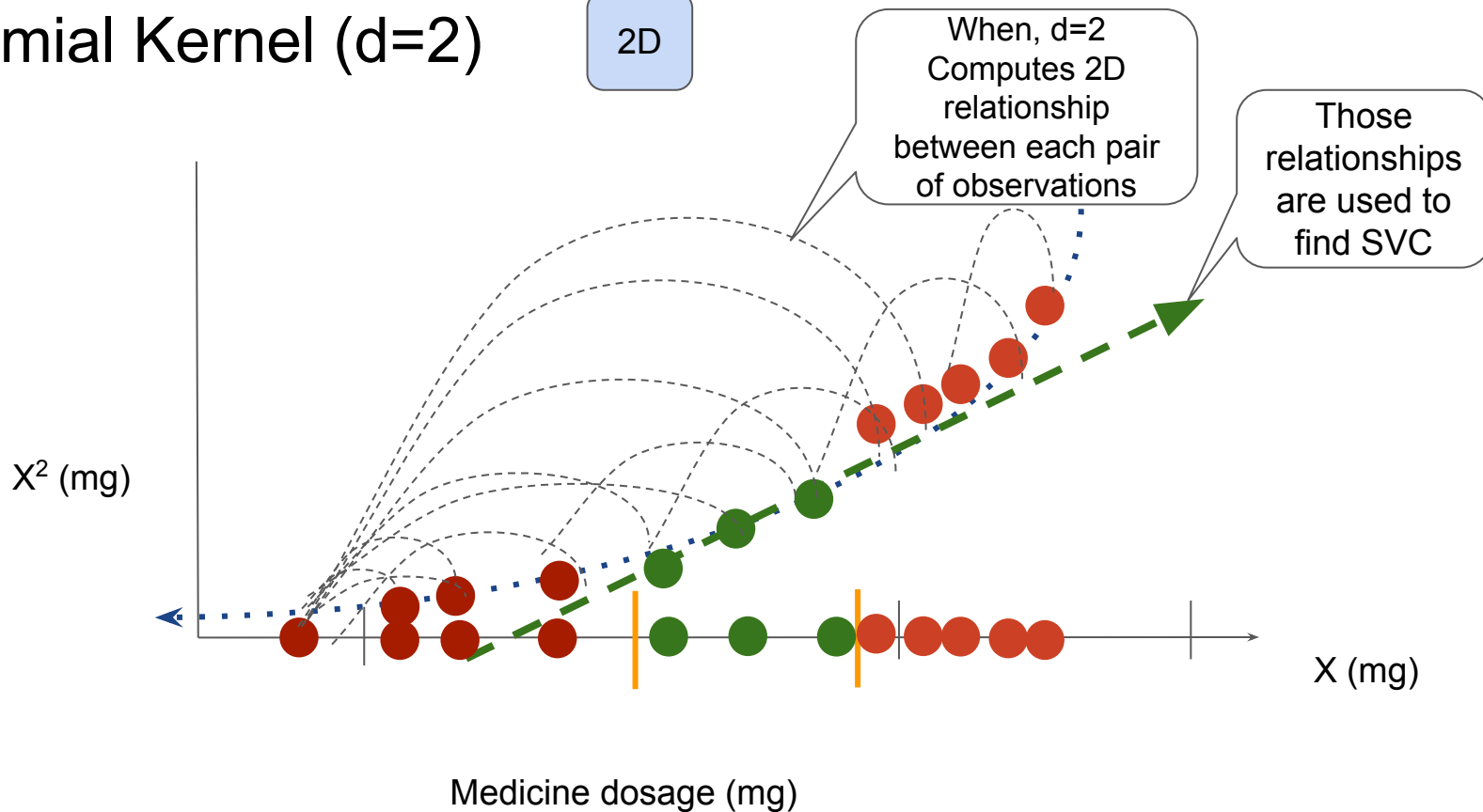


Polynomial Kernel has a parameter  $d$ , which stands for the degree of the polynomial.

When  $d=1$ , the polynomial kernel computes the relationship between each pair of the observations in 1D

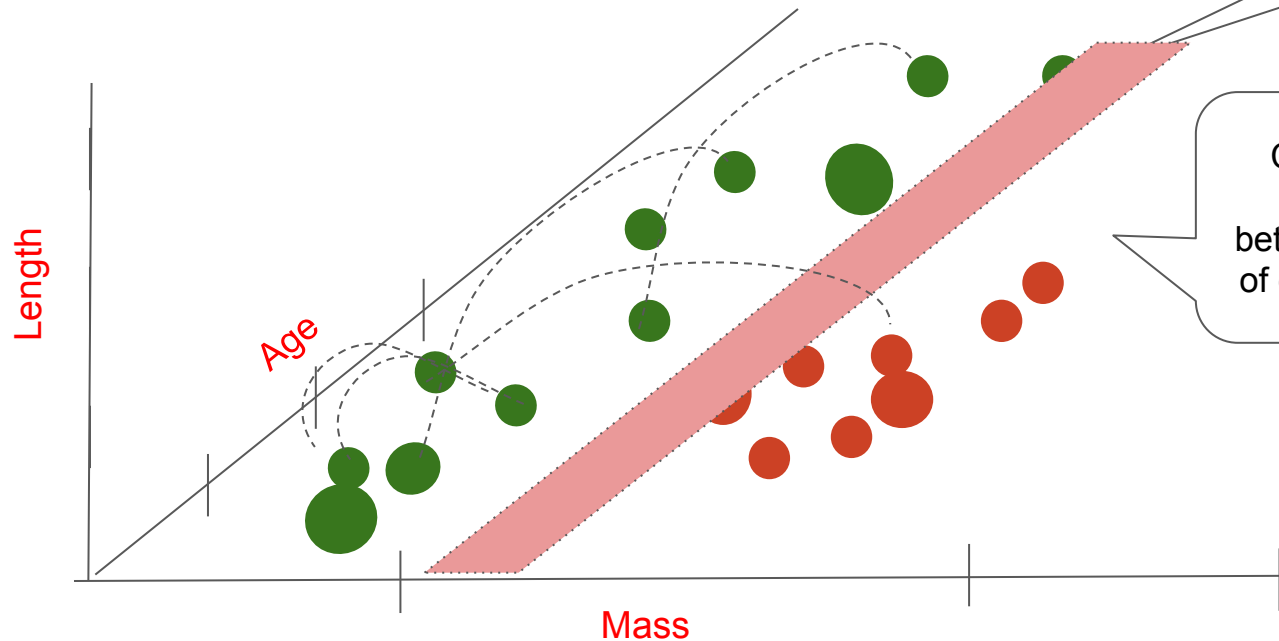
# Polynomial Kernel (d=2)

2D



# Polynomial Kernel (d=3)

3D

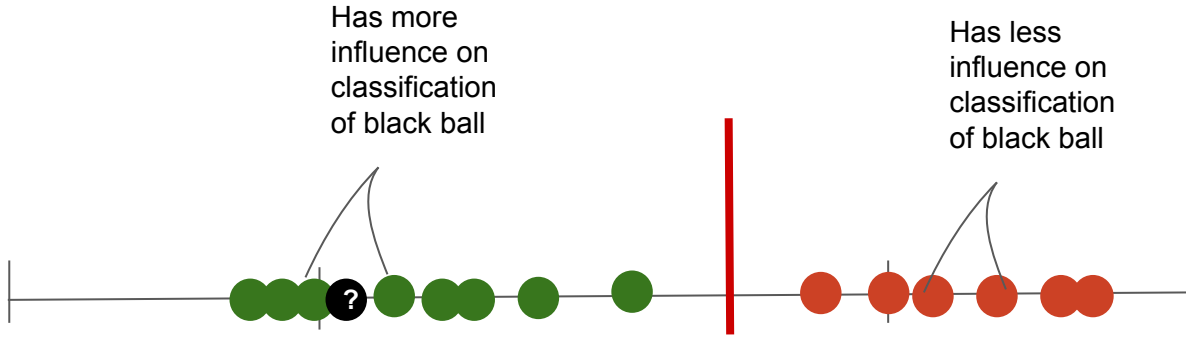


Maximal  
margin  
classifier  
(Plane)

When,  $d=3$   
Computes 3D  
relationship  
between each pair  
of observations to  
find SVC

Find  
suitable  $d$   
by cross  
validation

# Radial Basis Function



Finds SVC  
in infinite  
dimensions

Behaves like  
weighted  
nearest  
neighbor model



# The Kernel trick

Although previous examples show the data being transformed from relatively low dimension to higher dimension, **kernel functions only calculate the relationships between each pair of points as if they are in the higher dimension**; they don't actually do the transformation. This trick is called **"The kernel trick"**.  
This reduces the amount of computation

# The “Kernel Trick”

- The linear classifier relies on an inner product between vectors  
 $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- If every datapoint is mapped into high-dimensional space via some transformation  $\Phi: \mathbf{x} \rightarrow \phi(\mathbf{x})$ , the inner product becomes:  
 $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$
- A *kernel function* is some function that corresponds to an inner product in some expanded feature space.

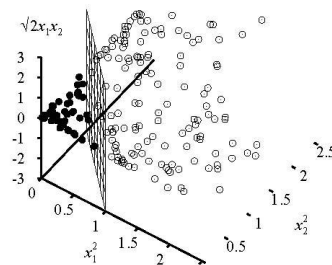
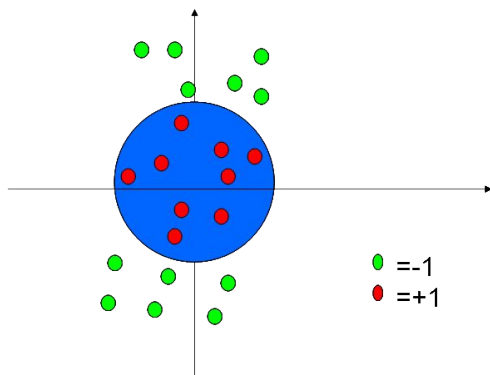
- Example:

2-dimensional vectors  $\mathbf{x} = [x_1 \ x_2]$ ; let  $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$ ,

Need to show that  $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ :

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= (1 + \mathbf{x}_i^T \mathbf{x}_j)^2 = 1 + x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{j1} x_{i2} x_{j2} + x_{i2}^2 x_{j2}^2 + 2 x_{i1} x_{j1} + 2 x_{i2} x_{j2} = \\ &= [1 \ x_{i1}^2 \ \sqrt{2} x_{i1} x_{i2} \ x_{i2}^2 \ \sqrt{2} x_{i1} \ \sqrt{2} x_{i2}]^T [1 \ x_{j1}^2 \ \sqrt{2} x_{j1} x_{j2} \ x_{j2}^2 \ \sqrt{2} x_{j1} \ \sqrt{2} x_{j2}] \\ &= \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \quad \text{where } \phi(\mathbf{x}) = [1 \ x_1^2 \ \sqrt{2} x_1 x_2 \ x_2^2 \ \sqrt{2} x_1 \ \sqrt{2} x_2] \end{aligned}$$

# Kernel Trick



Data points are linearly separable  
in the space  $(x_1^2, x_2^2, \sqrt{2}x_1x_2)$

We want to maximize  $\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \langle F(\mathbf{x}_i) \cdot F(\mathbf{x}_j) \rangle$

Define  $K(\mathbf{x}_i, \mathbf{x}_j) = \langle F(\mathbf{x}_i) \cdot F(\mathbf{x}_j) \rangle$

Cool thing :  $K$  is often easy to compute directly! Here,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle^2$$

# Questions?



# Contents

Machine Learning intro

Supervised vs unsupervised learning

Bias

Variance

Cross validation

Support vectors

Support vector machines

Kernel function

# Reference

<https://www.youtube.com/watch?v=efR1C6CvhmE>