Support Vector Machines (SVM)

Session-11

CLARUSWAY©
WAY TO REINVENT YOURSELF

SUMMARY of PREVIOUS CLASS

- **KNN**

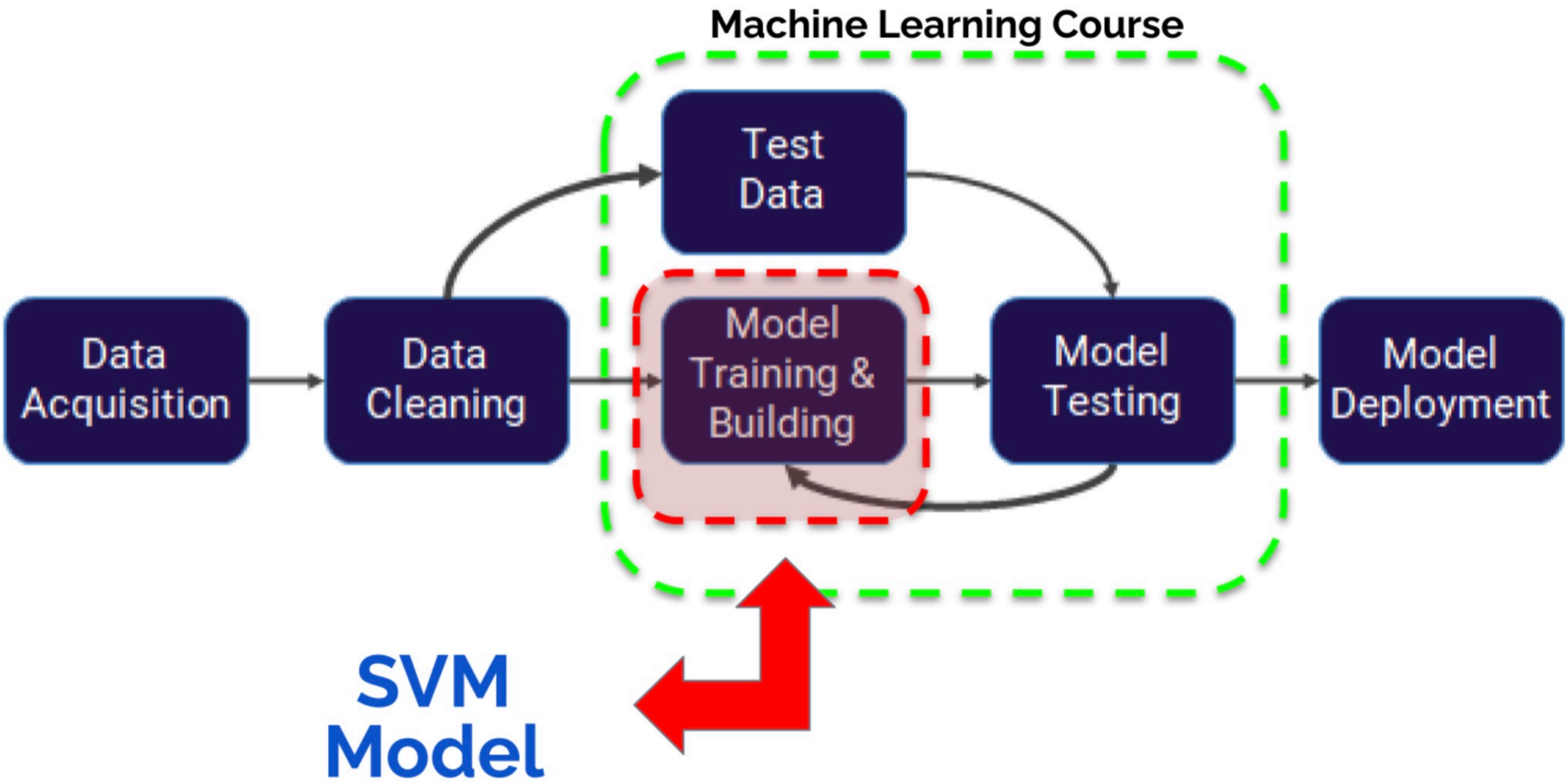
- **Selection of K:** **Elbow, Gridsearch**
- **Distance:**
  - Euclidian
  - Manhattan
  - Minkowski
- **Weights parameters**
  - Uniform: Majority voting
  - Distance: Weighted majority voting
- **Data Scaling is important**

Support Vector Machines Theory

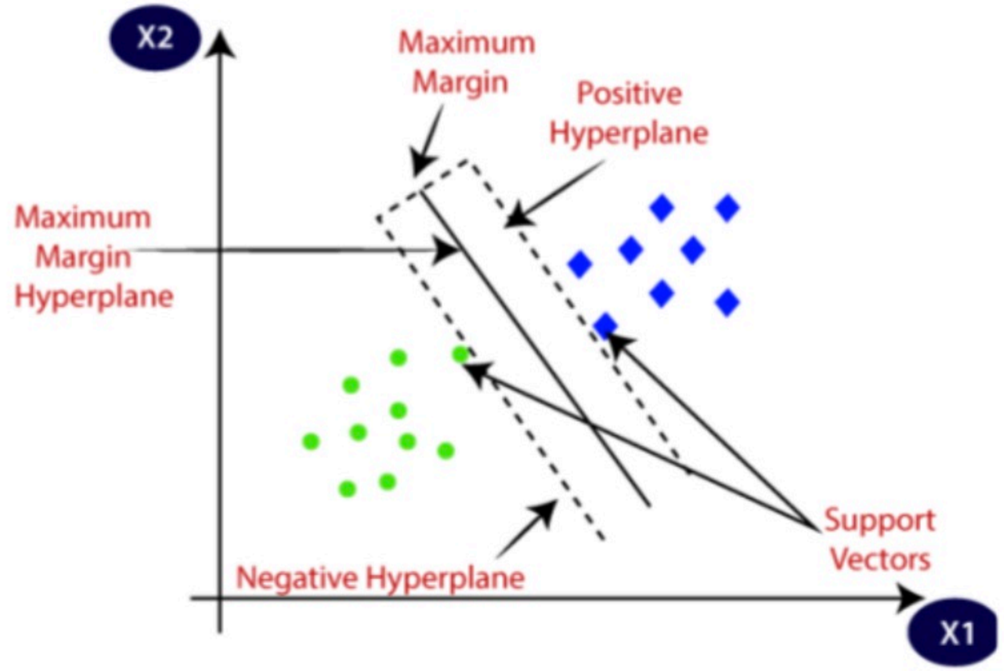
Support Vector Machines with Python

CLARUSWAY©
WAY TO REINVENT YOURSELF

Where are we?


Machine Learning Course
Test Data
Data Acquisition
Data Cleaning
Model Training & Building
Model Testing
Model Deployment
SVM Model

SVM Theory



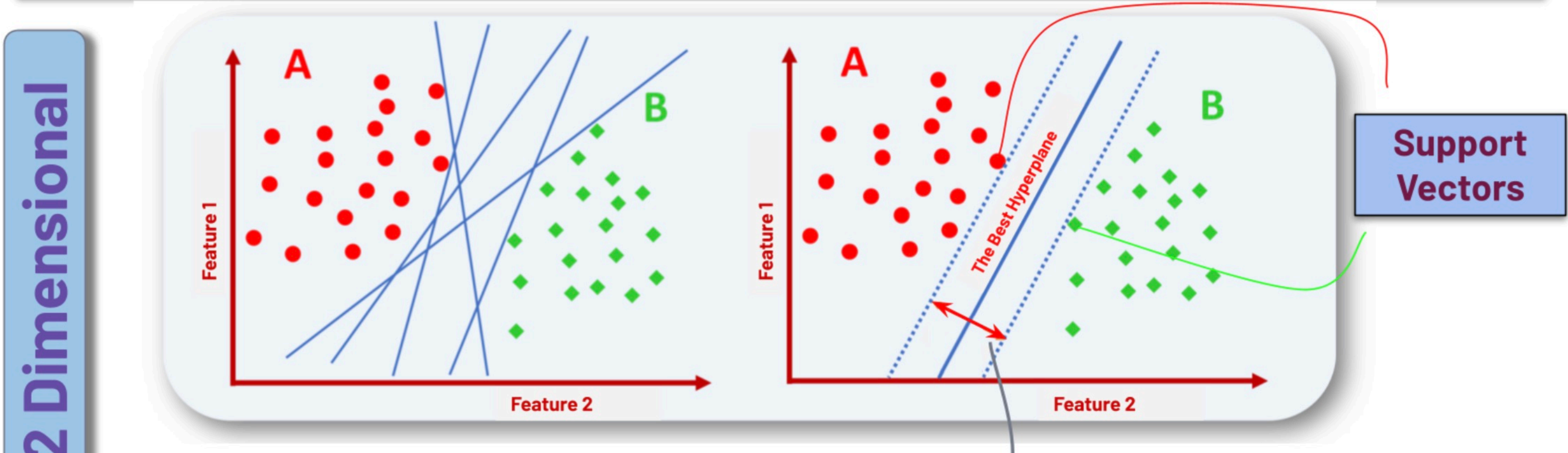
Support Vector Machines are used especially in **classification problems** with **high accuracy** capacity.

Some usage areas of SVM classification problems:

* Text Classification,
* Pattern Recognition,
* Image-based Gender Detection etc.

SVM Theory

The main goal of the SVM algorithm is to **find a hyperplane** in the space of variables that perfectly separates classes.

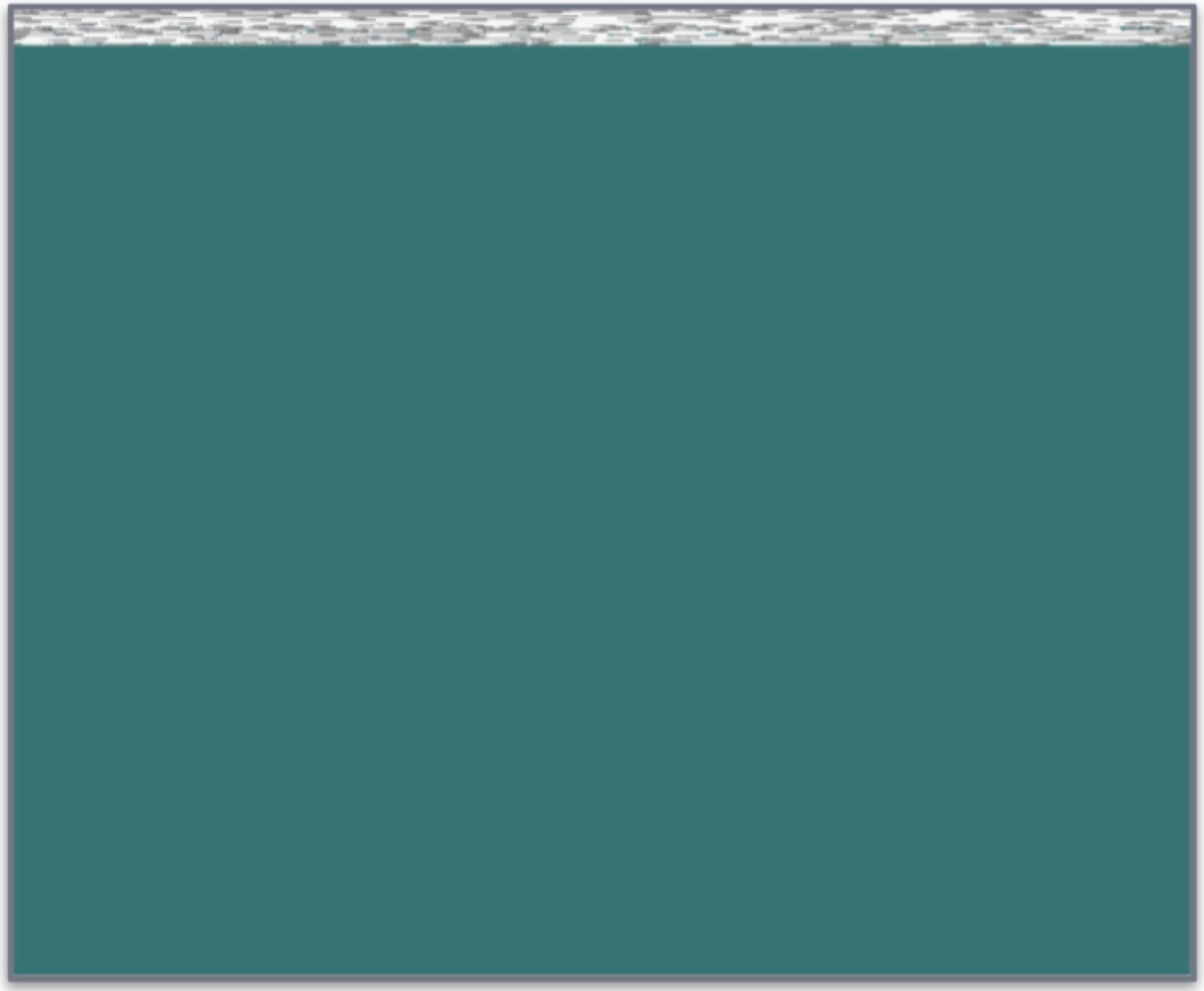


SVM picks the best separating hyperplane based on the **maximum margin (Max. Margin Classification)**

SVM Theory

We can think of **hyperplanes as <u>decision boundaries</u>** founded using classification.

3 Dimensional



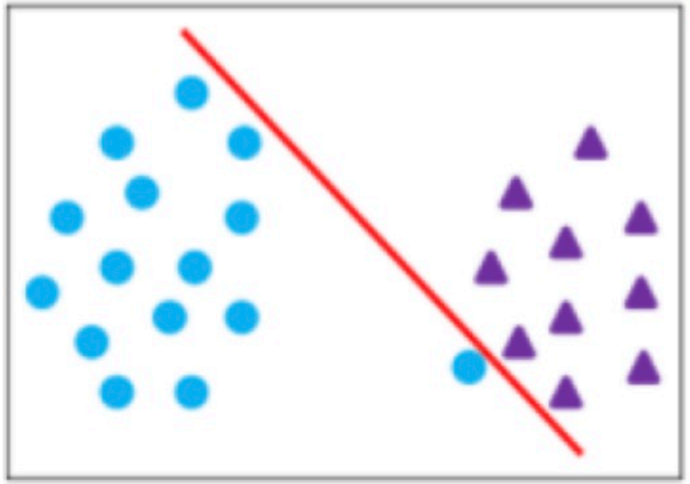
Since we have **two variables**, our hyperplane was **just one line**.

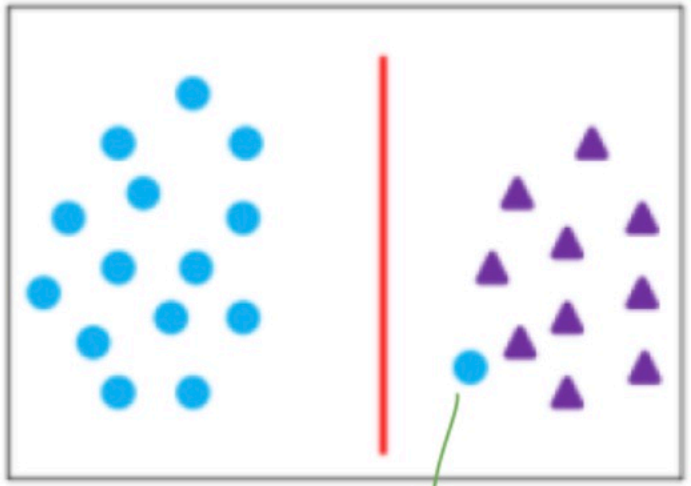
We see the **hyperplane** created for a **three variable dataset**.

However, when the number of variables exceeds three, we will not be able to show them visually.

SVM Theory

Soft Margin Classification





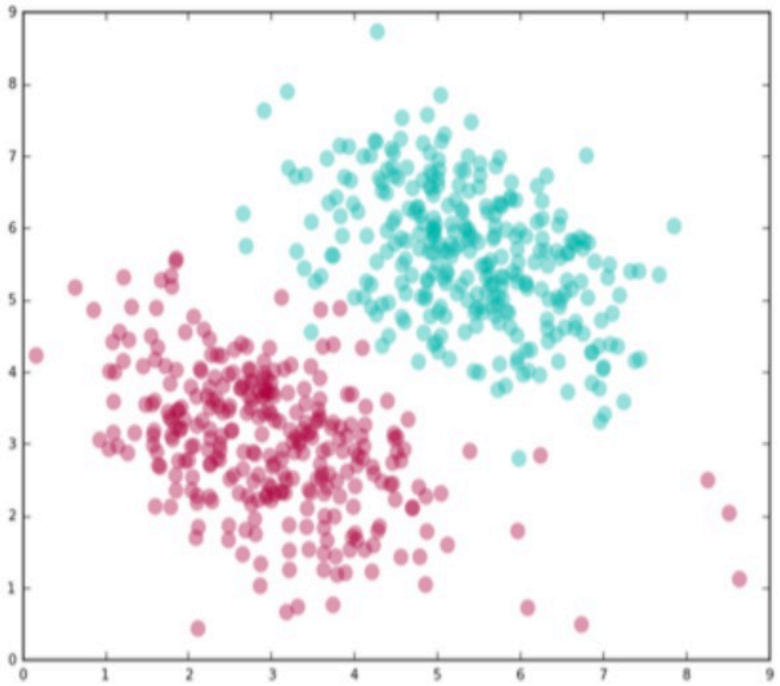
Data linearly separable by only a **very narrow margin.** **(Hard Margin)**

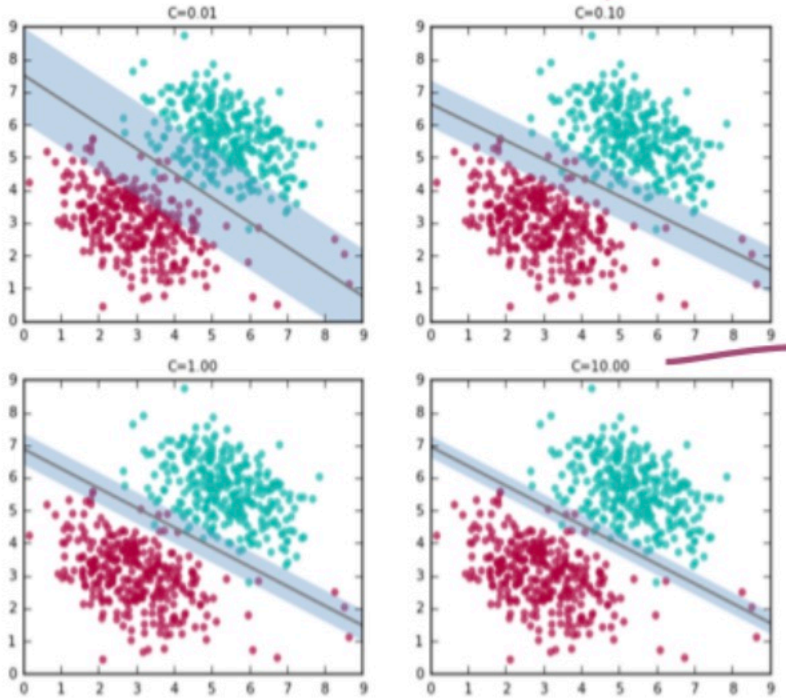
Large margin separation **is better** in case of **minor misclassifications.** **(Soft Margin)**

* **Hard margin classification** requires that all data points are classified correctly.

* **Soft margin classification** allow <u>misclassification</u> on noisy data points.

SVM Theory



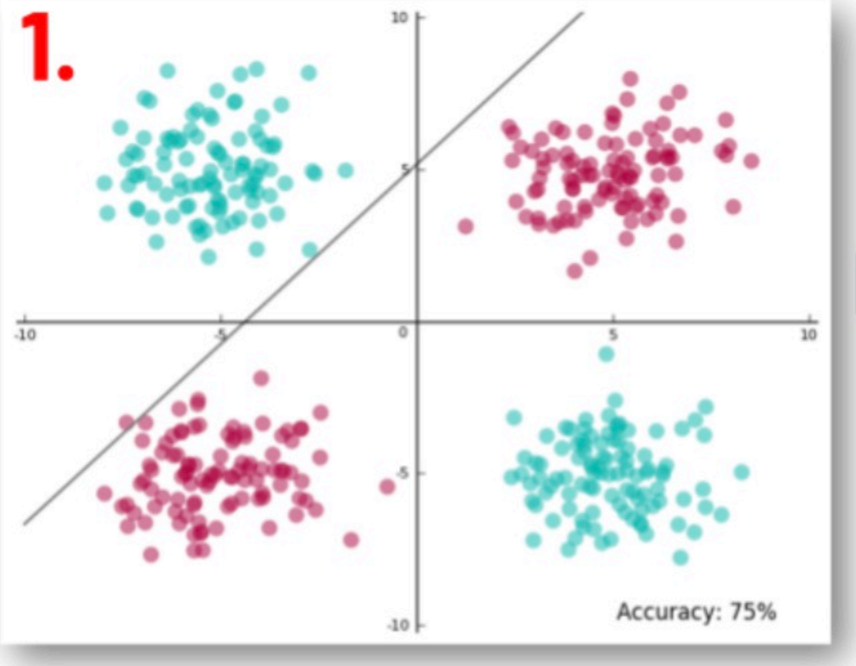


"C" parameter is the **regularization** parameter of the error.

* Although there are very close values above, we see a data set that can be separated quite successfully linearly.
* We can estimate the approximate line although there are **some errors**.
* With the changes in the **"C" parameter** we can change the margin size and **determine how much error we have tolerance**.

SVM Theory

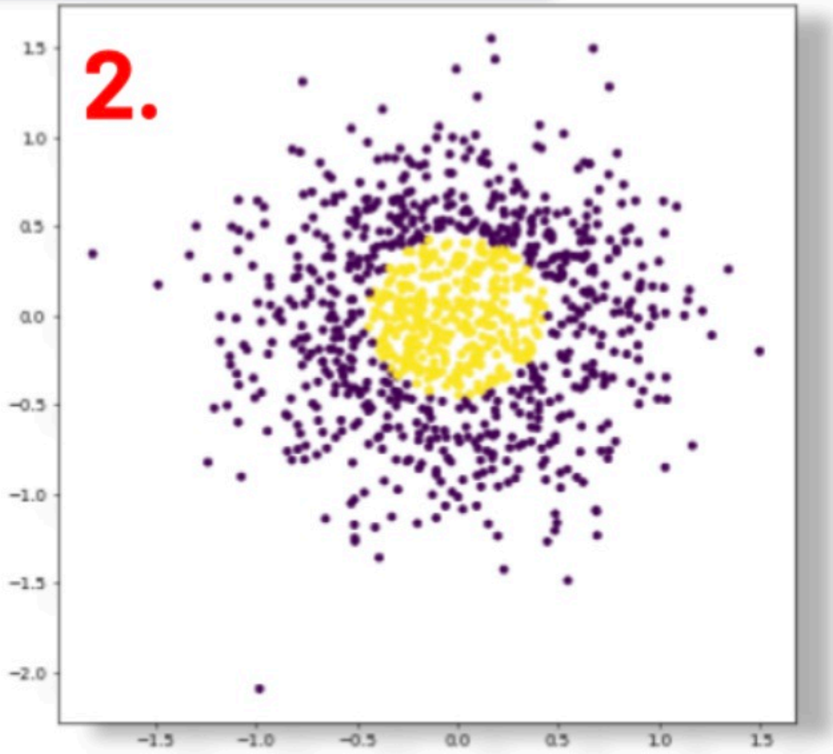
What if the data is non linearly divisible?



?

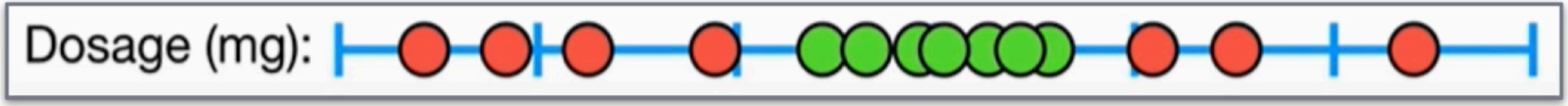
How do we separate the classes ?

?



?

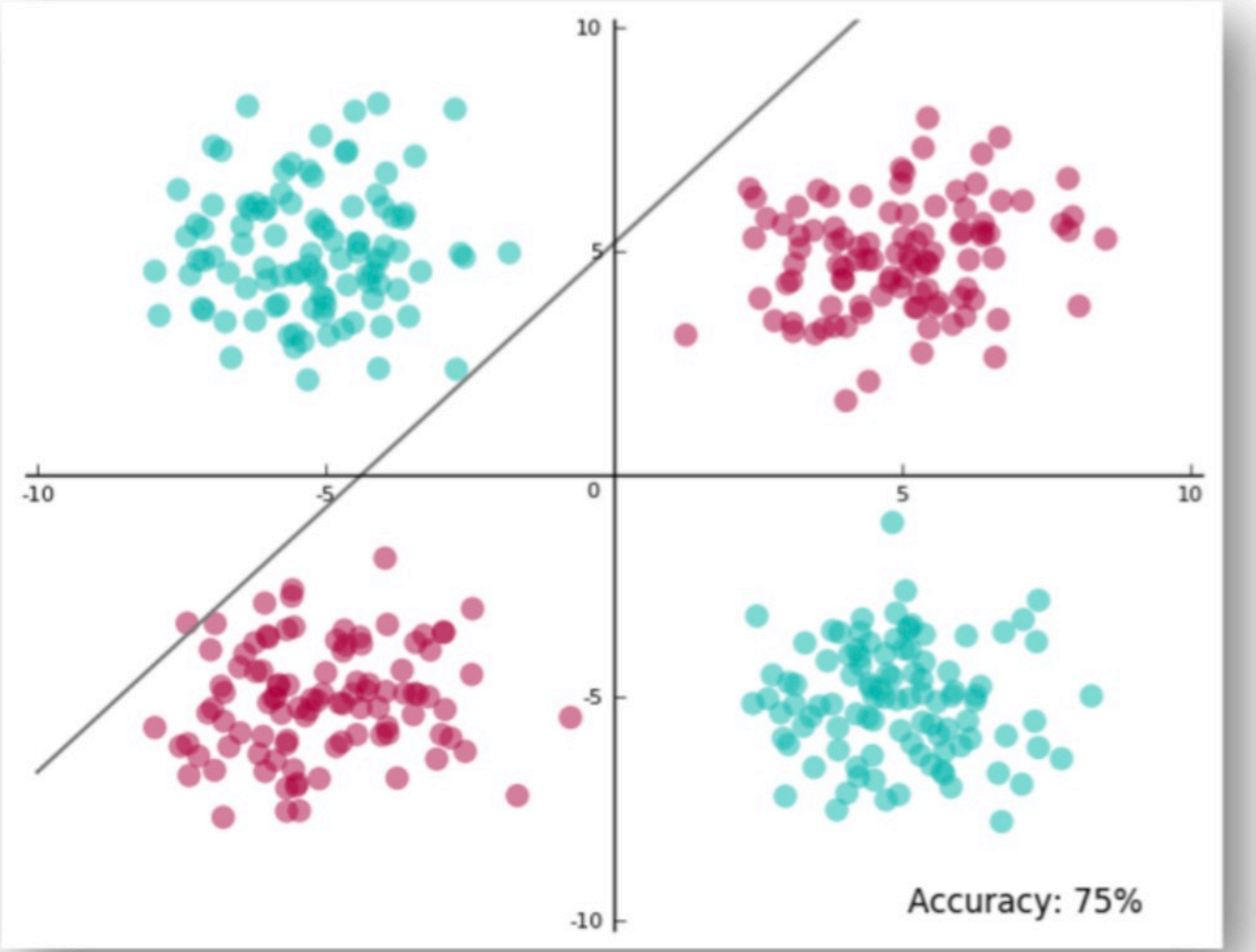
3.



KERNEL TRICK

SVM Theory

Sample-1



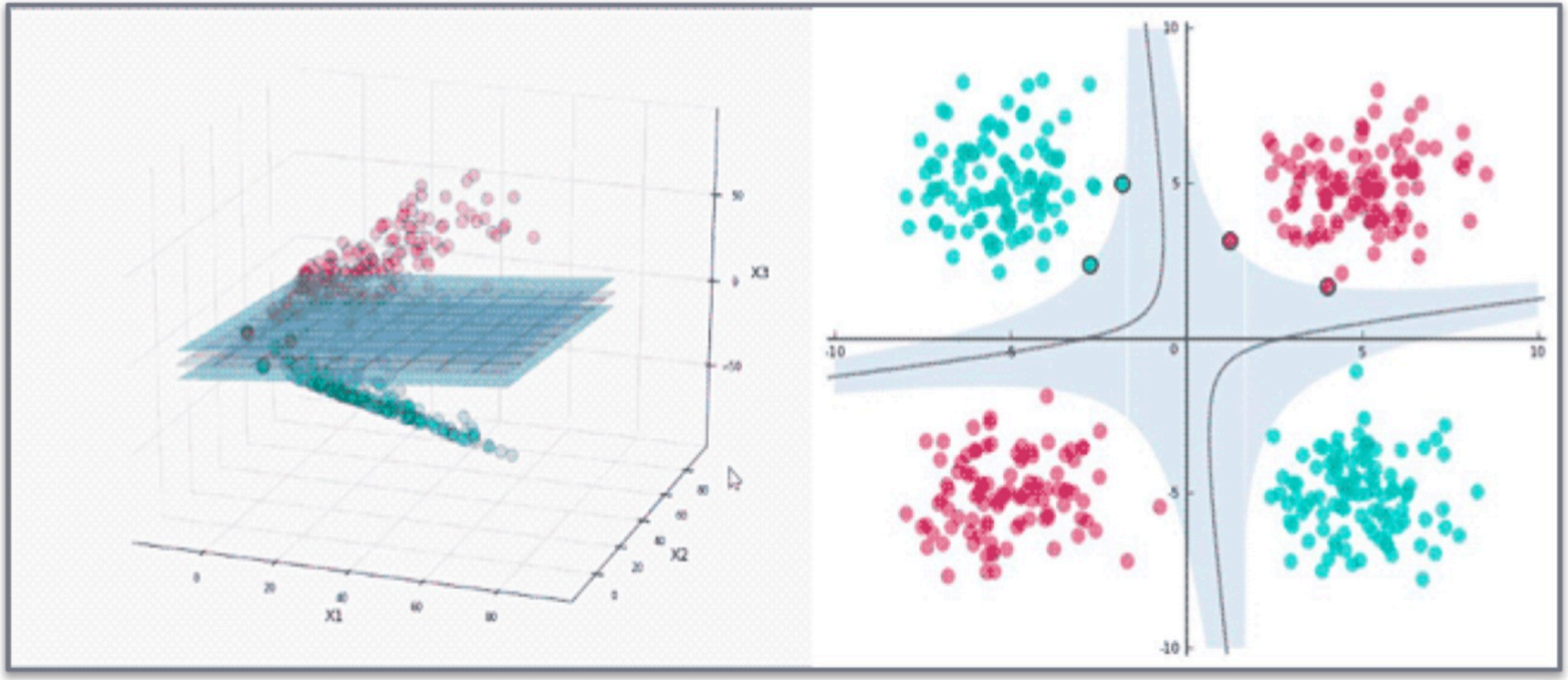
If a hyperplane is selected like the example, the success of the model **will <u>not exceed 75%.</u>**

If we make our **2-dimensional dataset to 3-dimensional**, we can divide our data with a **more uniform hyperplane.**

SVM Theory

Sample-1

When we **square** our variables **(2 D)** and **produce a new variable (3 D)**, our data will become better separable.



2D to 3D

$$X_1 = x_1^2$$

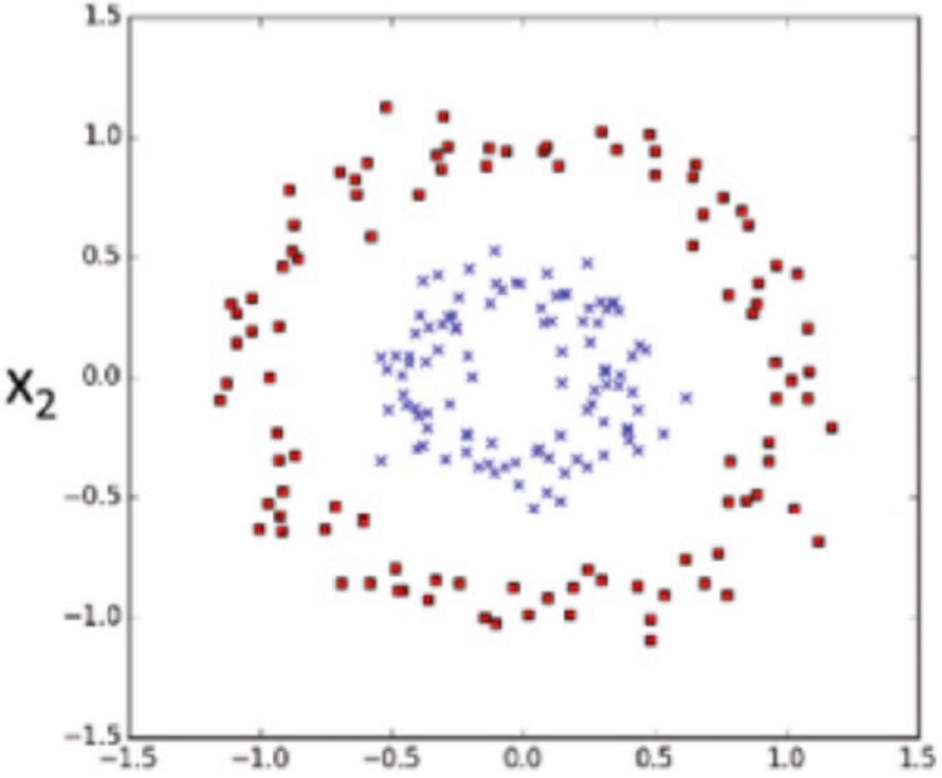
$$X_2 = x_2^2$$

$$X_3 = \sqrt{2x_1x_2}$$

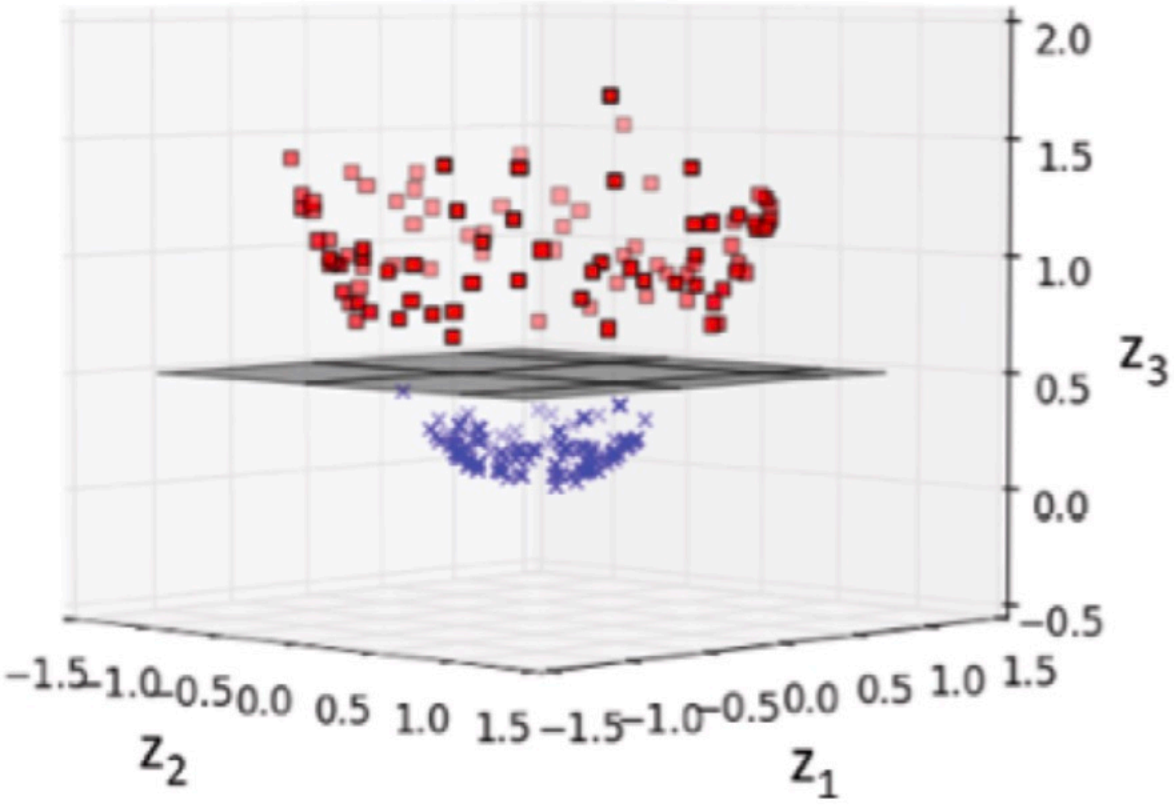
Source: KDnuggets

SVM Theory

Sample-2



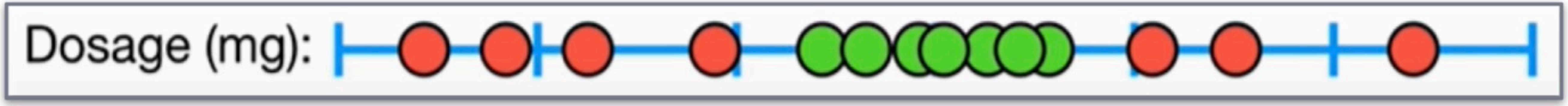
2D to 3D

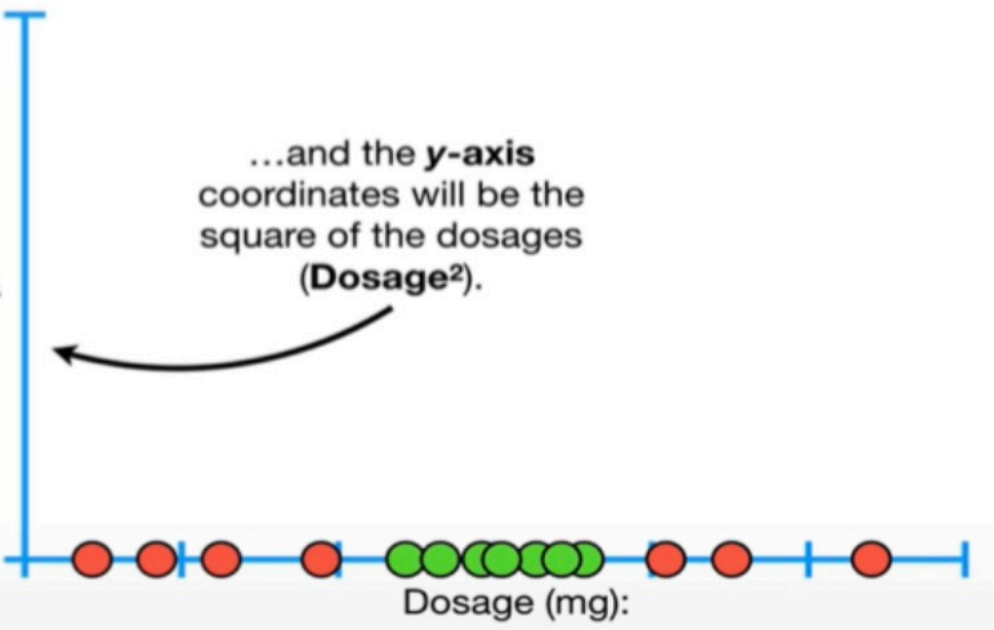


SVM Theory

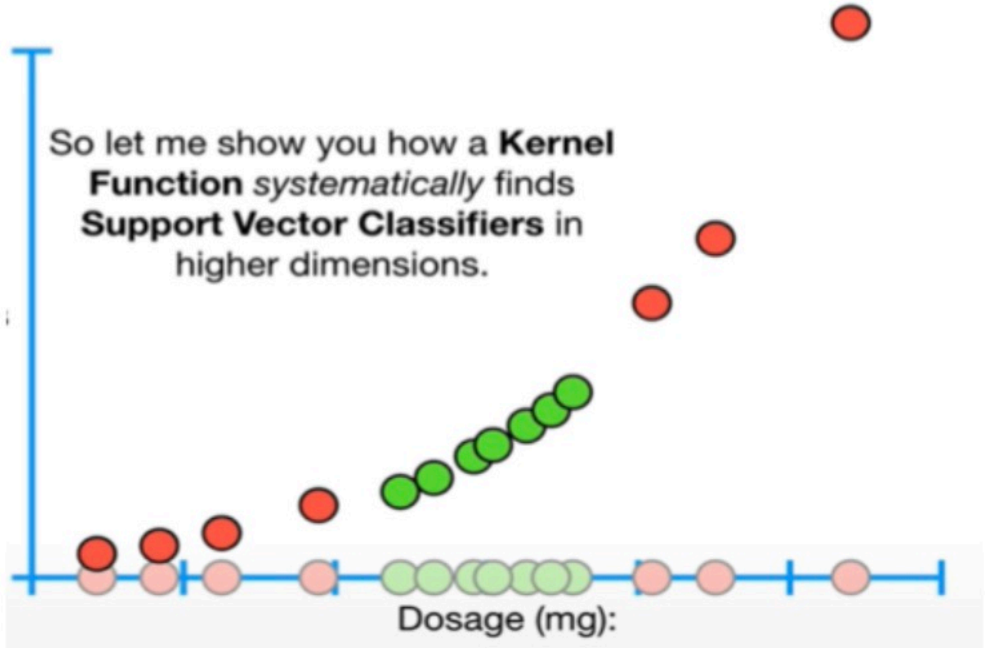
Sample-3

1.

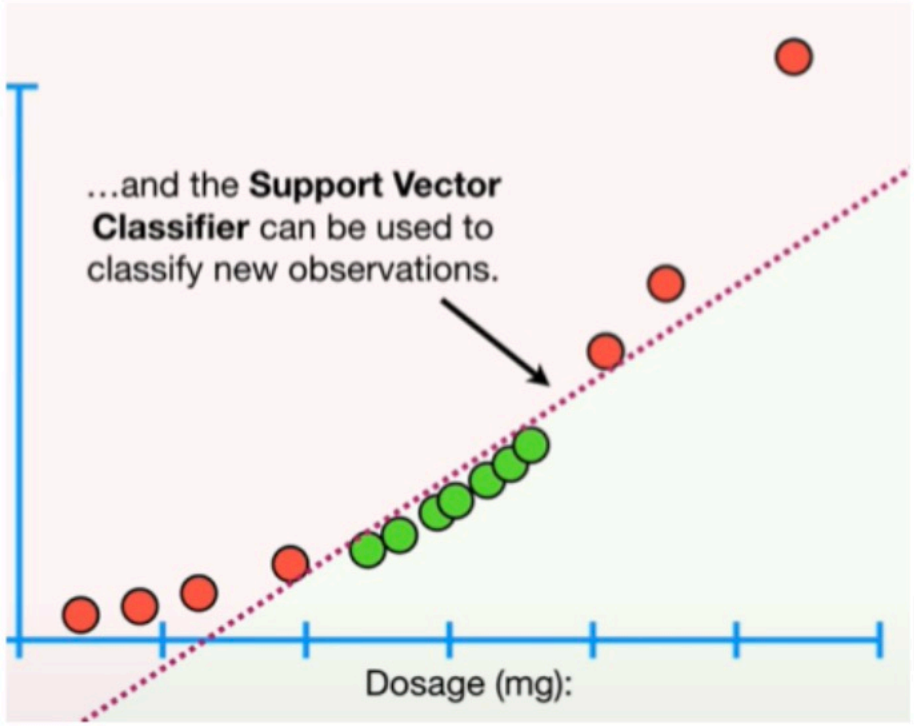




2. New *axis (y)*



3. 1D to 2D



4. Hyperplane

Sample from Statquest

SVM Theory

Different Kernel Functions of SVM

	Formula	Parameters
Linear	$K(x, x_i) = x \cdot x_i$	/
Polynomial	$K(x, x_i) = [\gamma * (x \cdot x_i) + coef]^d$	$\gamma, coef, d$
RBF	$K(x, x_i) = \exp(-\gamma * \|x - x_i\|^2)$	$\gamma.$
Sigmoid	$K(x, x_i) = \tanh(\gamma(x \cdot x_i) + coef)$	$\gamma, coef$

Source: http://scikit-learn.org/stable/auto_examples/svm/plot_iris.html

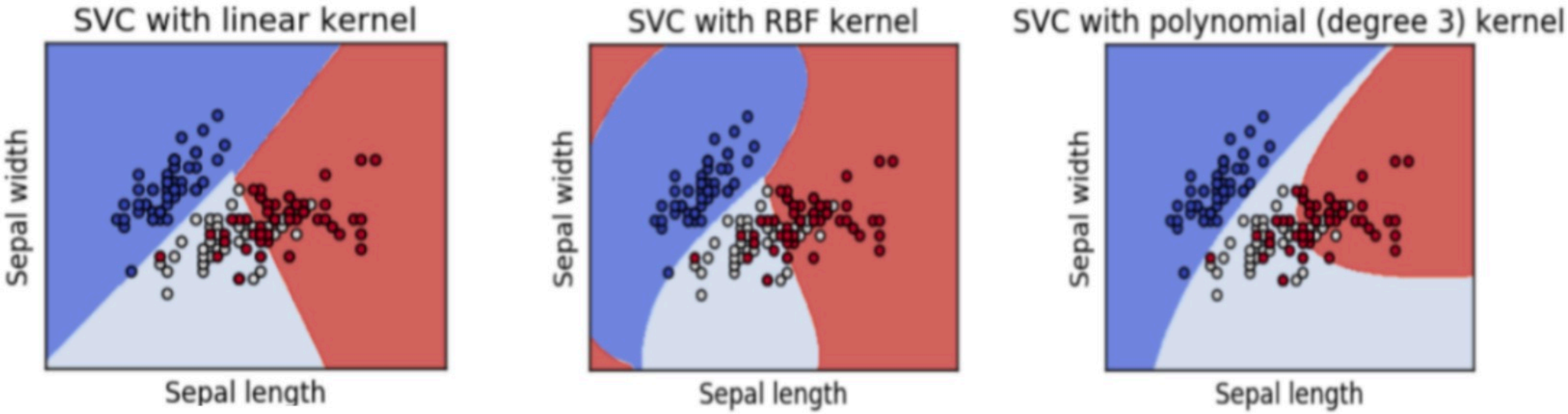
SVM Theory

SVM Algorithm:

Hyperparameters

"kernel" ***parameter:*** **(default="rbf")**
{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}

Specifies the kernel type to be used in the algorithm.



Source: http://scikit-learn.org/stable/auto_examples/svm/plot_iris.html

SVM Theory

SVM Algorithm:

Hyperparameters

“kernel” *parameter:* (default = “rbf”)
{‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’}

What to use when?

- A **linear SVM** can perform well for problems with **many features** but **limited number of instances,**
- A **rbf kernel (default)** is good for **complex, nonlinear problems** where the number of **features is small and number of instances is large.**

SVM Theory

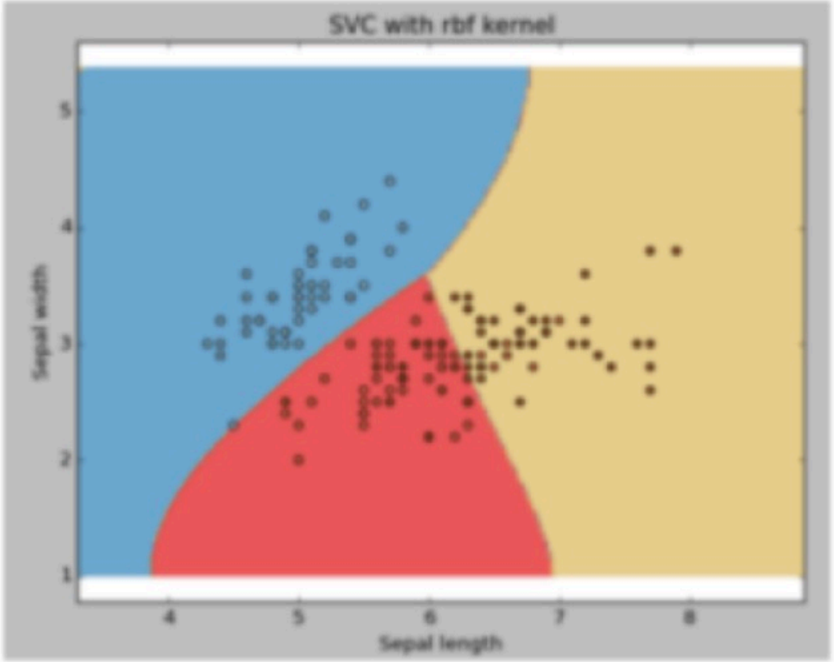
SVM Algorithm:

Hyperparameters

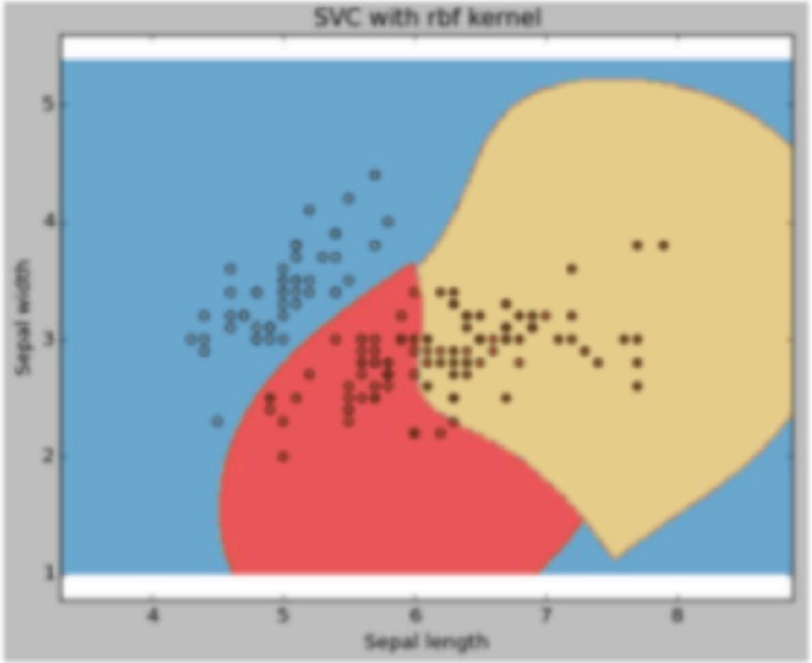
"C" ***parameter:*** **(default = 1)**

Penalty parameter of the error term.

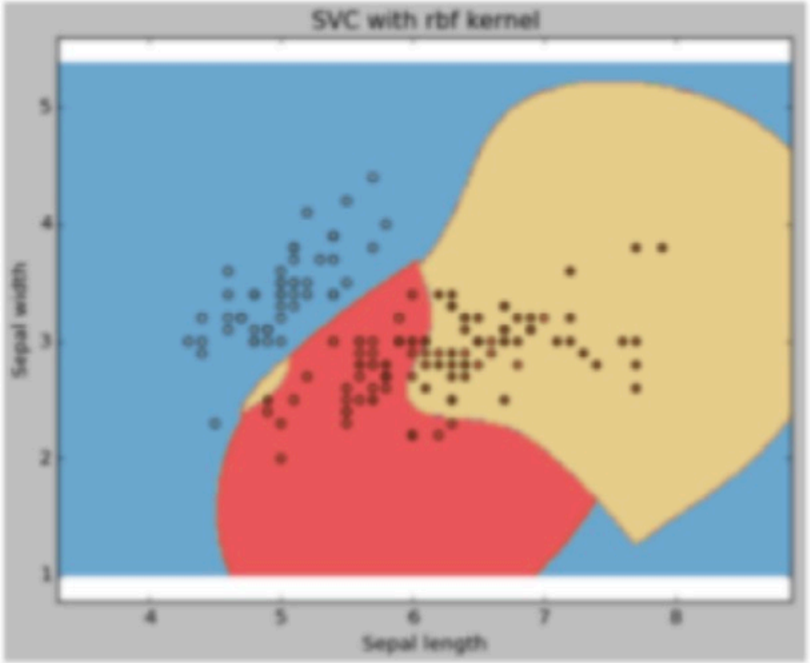
c =1 (soft/large margin hyperplane)



C =100



c =1000 (hard margin)



SVM Theory

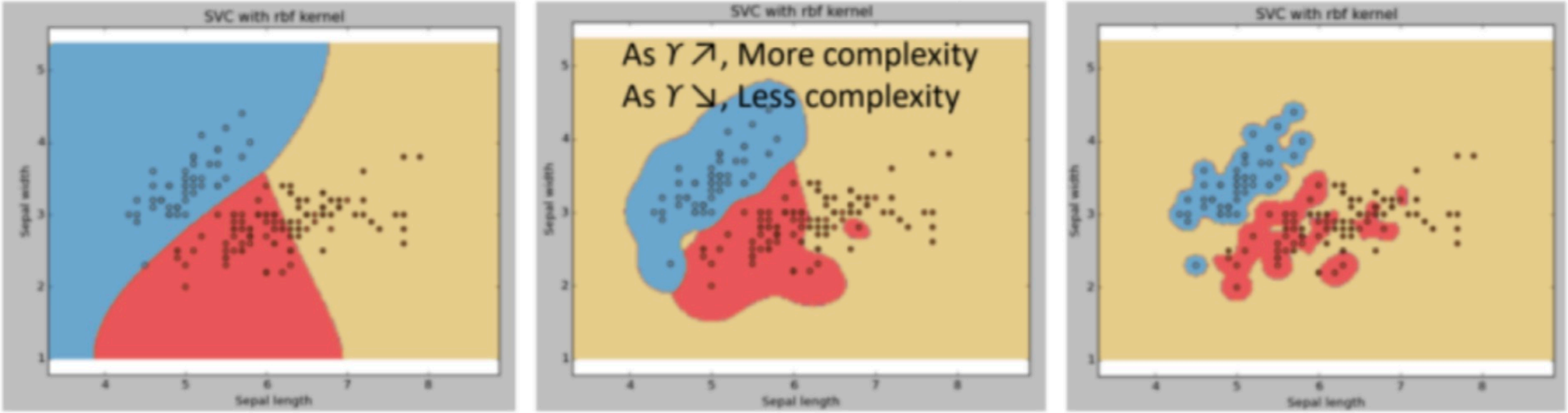
SVM Algorithm:

Hyperparameters

“gamma” ***parameter:*** **(default = 1)**

Kernel coefficient for ‘**rbf**’, ‘**poly**’ and ‘**sigmoid**’.

gamma =0 | gamma =10 | gamma =100



if gamma up, overfit possibility up

SVM Theory

Pros & Cons

Pros:

– Works great with Computer Vision problems

– Low generalization error

– Can model complex relationships (powerful & flexible kernels)

Cons:

– Need parameter tuning and sensitive to kernel choice

– Sensitive to noise in the data