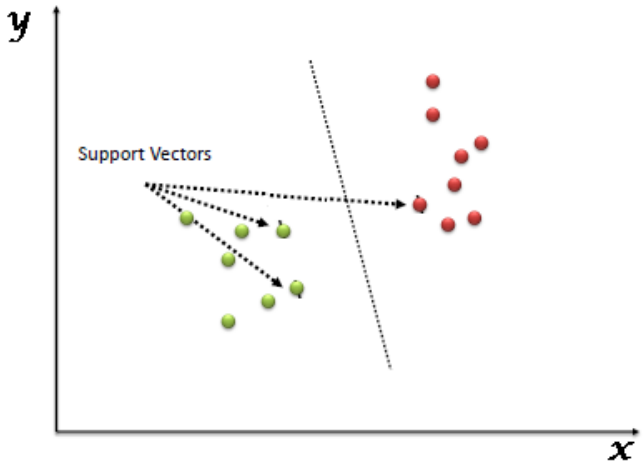
# Support Vector Machine

# SVM

- **Support Vector Machine** (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems.
- In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate.
- Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).
- **Support Vectors are simply the coordinates** of individual observation. The **SVM classifier is a frontier** that best segregates the **two classes (hyper-plane/ line).**

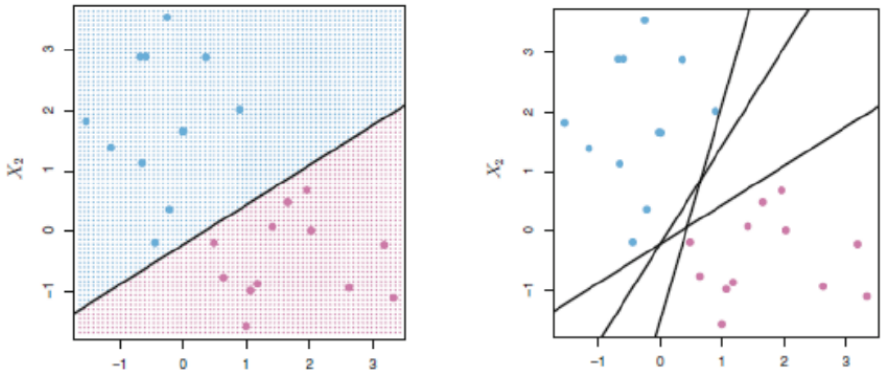


# SVM Example

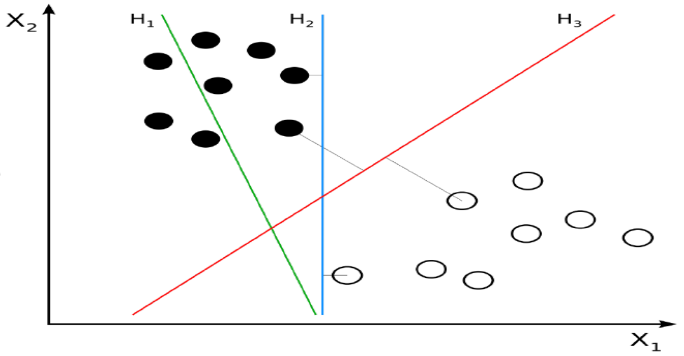
- Suppose that we have a dataset that has X1 and X2 as independent features and Category as a dependent feature.

| X1 | X2 | Category |
|---|---|---|
| 60 | 82 | Pass |
| 20 | 42 | Fail |
| ... | ... | ... |
| 91 | 72 | Pass |

- There are many possibilities to separate the two category classes – But which is best among all?



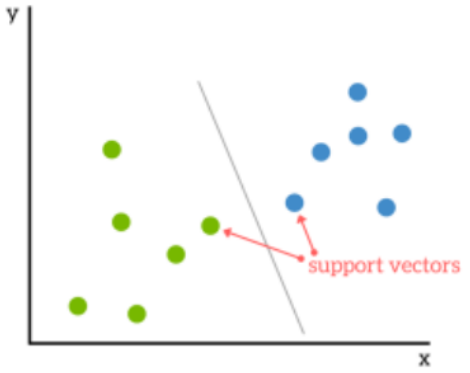
- So, among these different hyperplanes, which is the best hyperplane? we can clearly see like among three-line, red line (H3) which has the maximum margin with the data points and also classified between the data properly, if you see blue color line (H2) the margin is small with one data and large with another data, whereas, green line (H1), it has not classified between the data itself.



# Support Vectors

## Support Vectors

- The data/vector points closest to the hyperplane (black line) are known as the support vector (SV) data points because only these two points are contributing to the result of the algorithm (SVM), other points are not.
- If a data point is not an SV, removing it has no effect on the model.
- Deleting the SV will then change the position of the hyperplane.
- The dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line.
- If the number of input features is 3, then the hyperplane becomes a two-dimensional plane.
- It becomes difficult to imagine when the number of features exceeds 3.



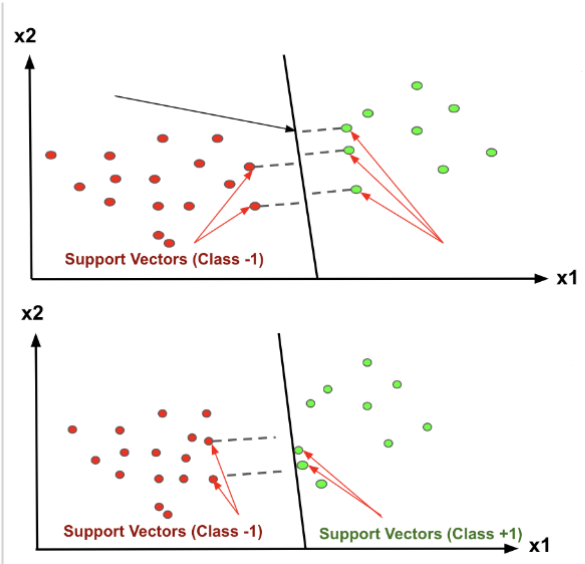
# SVM vs SVC

## How Support vector machines are different than Support vector classifier?

We often get confused with the terms SVM and SVC, the simple answer is if **the hyperplane** that we are using for classification **is in linear condition**, then the condition is **SVC.**

If **the hyperplane** that we are using for classification **is in non linear condition**, then the condition is **SVM.**

The **distance of the vectors from the hyperplane** is called the **margin** which is a separation of a line to the closest class points. We would like to choose a hyperplane that maximizes the margin between classes.




**Good Margin**

- all sector vectors have the same distance with the maximum margin hyperplane

**Bad Margin**

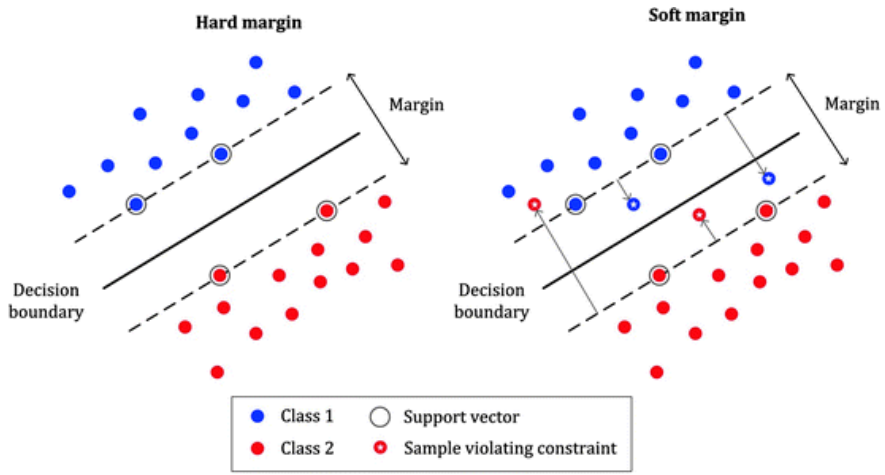
- very close to either class -1 support vectors or class +1 support vectors

# Soft Margin vs Hard Margin

## Margins in SVM:

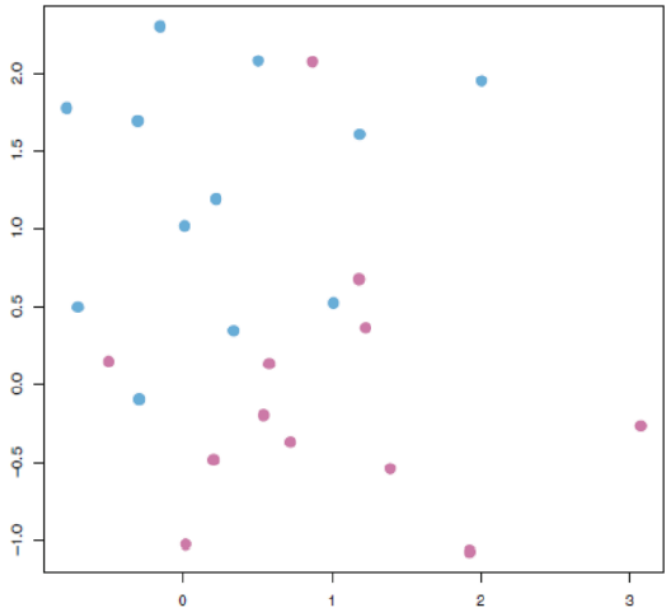
Margin can be sub-divided into,

1. **Soft Margin** – As most of the real-world data are not fully linearly separable, **we will allow some margin violation** to occur **which is called soft margin classification**. It is better to have a large margin, even though some constraints are violated. Margin violation means choosing a hyperplane, which can allow some data points to stay on either the incorrect side of the hyperplane and between the margin and correct side of the hyperplane.
2. **Hard Margin** – If the training data is linearly separable, we can select two parallel hyperplanes that separate the two classes of data, so that the distance between them is as large as possible.



# Limitations of SVC

## Inefficient in case of Non linear data



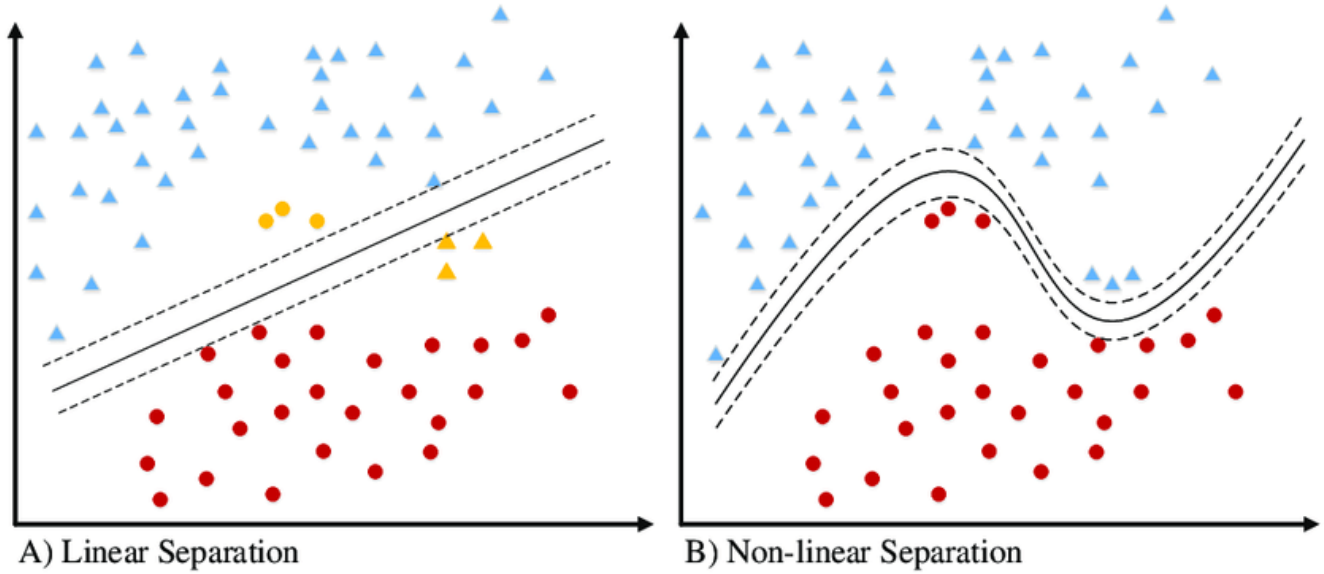

Here in the above picture if we notice clearly, we cannot draw a hyperplane in this scattered data to separate the data points between classes (for classification!) by a straight line.

Therefore SVC is not efficient to classify the observations in case of non linear data.

# Support vector machines

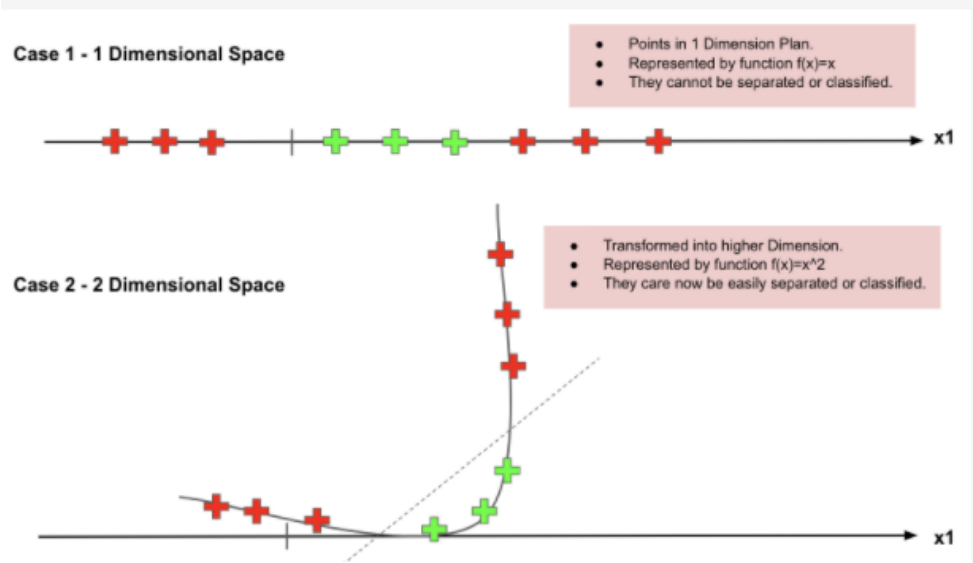
The limitation of SVC is compensated by SVM non-linearly. And that's the difference between SVM and SVC.

If the hyperplane classifies the dataset linearly then the algorithm we call it as SVC and the algorithm that separates the dataset by non-linear approach then we call it as SVM.



# Kernel Trick

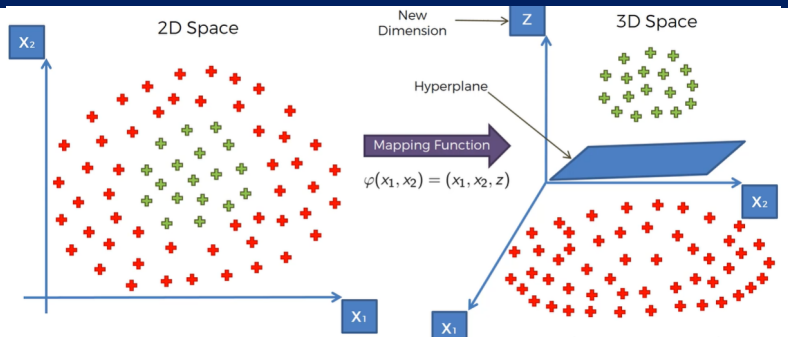
- SVM has a technique called the kernel trick.
- These are functions that take low dimensional input space and transform it into a higher-dimensional space i.e. it converts not separable problem to separable problem.
- It is mostly useful in non-linear separation problems.



# Steps of Non linear transformation

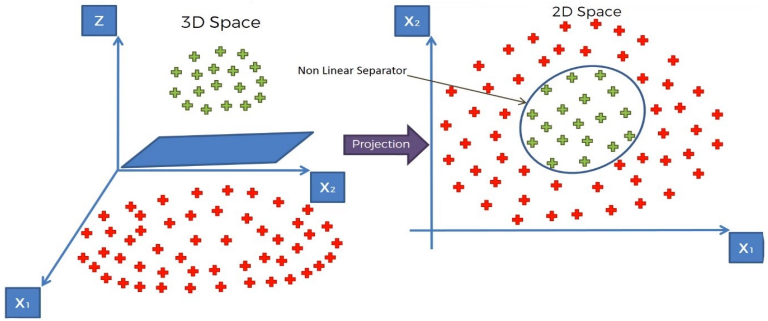
## Step 1

Data points which is not able to linearly classified is converted into higher dimensional, then get separated linearly in higher space




## Step 2

After separation in higher dimensions, data points are non-linearly separated in the original dimension of lower space.



# Kernels in SVM

**Non-linear kernels in SVM are:**

1. Polynomial SVM Kernel
2. Gaussian Radial Basis Function (RBF)
3. Sigmoid Kernel

## Polynomial SVM Kernel

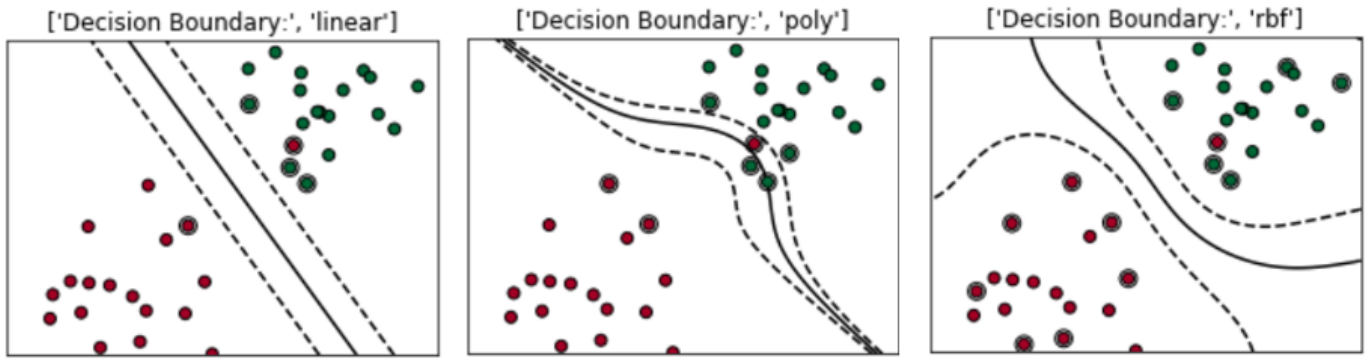
polynomial: $(\gamma\langle x, x'\rangle + r)^d$, where $d$ is specified by parameter `degree`, $r$ by `coef0`

## Radial Basis Function (RBF)

rbf: $\exp(-\gamma\|x - x'\|^2)$, where $\gamma$ is specified by parameter `gamma`, must be greater than 0.

## Sigmoid Kernel

sigmoid $\tanh(\gamma\langle x, x'\rangle + r)$, where $r$ is specified by `coef0`.



# Parameters in SVM

## Parameters of SVM in sklearn:

**kernel** : ***{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}***

**degree** : Degree of the polynomial kernel function ('poly'). Ignored by all other kernels.

**c** : ***float, default=1.0***

Inverse of regularization strength; must be a positive float. smaller values specify stronger regularization

**gamma** : ***{'scale', 'auto'} default='scale'***

Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.

if gamma='scale' (default) is passed then it uses 1 / (n_features * X.var()) as value of gamma,

if 'auto', uses 1 / n_features.

**coef0** : ***float, default=0.0***

Independent term in kernel function. It is only significant in 'poly' and 'sigmoid'.