# Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.



**SVM can be of two types:**

* **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
* **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

**Hyperplane:** There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

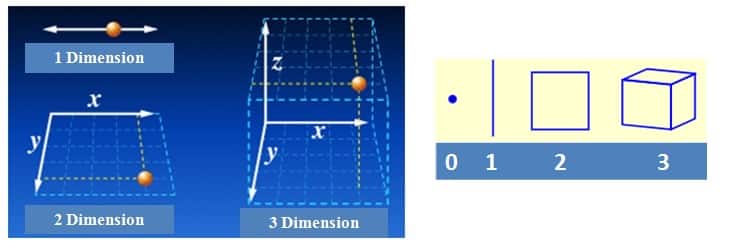
The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

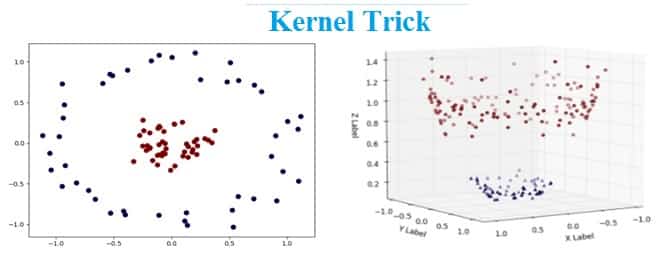
**Support Vectors:**

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

**Dimensions:** In simple terms, a dimension of something is a particular aspect of it. Examples: width, depth and height are dimensions. A line on a plane is one dimension, considering the edges a square has two dimensions and a cube has three dimensions.



**Kernel:** A kernel is a method of placing a two dimensional plane into a higher dimensional space, so that it is curved in the higher dimensional space. (In simple terms, a kernel is a function from the low dimensional space into a higher dimensional space.)

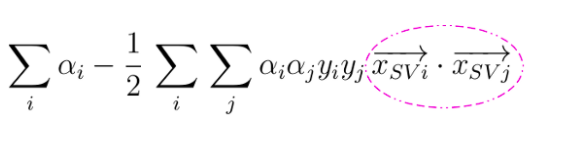




# Non-linear data set mapped to higher dimension

## **What is Kernel Trick?**

In order to deal with the above challenge of computing (expensive) the data coordinates (new features) using mapping function to transform to higher-dimensional space, the **kernel trick**is applied. **Kernel trick is** to convert dot product of support vectors to the dot product of mapping function. Recall the optimization problem for SVM:

**Fig 6: SVM Optimization – Objective Function**

Pay attention to the pink circle which represents the dot product of the support vectors. T**he trick is to replace this inner product with another one, without even knowing Φ**. In fact, there can be many different feature maps that correspond to the same inner product. The above equation will look something like the following after replacing the dot product of support vectors with the dot product of the mapping function.

# **Why is it important to use the kernel trick?**

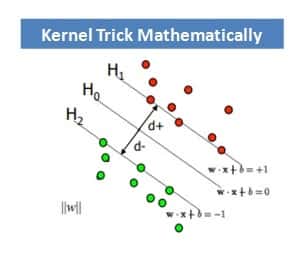
As you can see in the above picture, if we find a way to map the data from 2-dimensional space to 3-dimensional space, we will be able to find a decision surface that clearly divides between different classes. My first thought of this data transformation process is to map all the data point to a higher dimension (in this case, 3 dimension), find the boundary, and make the classification.

That sounds alright. However, when there are more and more dimensions, computations within that space become more and more expensive. This is when the kernel trick comes in. It allows us to operate in the original feature space without computing the coordinates of the data in a higher dimensional space.

## **The Kernel Trick Mathematically**

Our basic idea of SVM and Kernel trick is to find the plane which can separate, classify or split the data with maximum margin as possible. The margin is also called street width. The distance from a point〖(x〗\_0,y\_0) to a line: Ax+By+c = 0 is: |Ax\_0 + By\_0 +c|/sqrt(A2+B2), so, The distance between H0 and H1 is then: |w•x+b|/||w||=1/||w||, so The total distance between H1 and H2 is thus: 2/||w||

In order to maximize the margin, we thus need to minimize ||w||. With the condition that there are no data points between H1 and H2: xi •w+b ≥ +1 when yi =+1 xi •w+b ≤ –1 when yi =–1 Can be combined into: yi (xi •w) ≥ 1



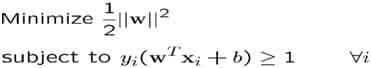
We can intuitively observe that we are trying to optimize the margin or street width by maximizing the distance between support vectors. An optimization problem typically consists of either maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function. Also we are trying to achieve the same with a constraint in mind where the support vectors should be away from the street and not on or in between the street. Hence we can say that this is a typical constrained optimization problem or situation.

This can be solved through Lagrangian formula or lagrangian multipliers. Let us first find the margin or decision boundary for linearly separable plane and then solve with Lagrange multipliers.

* The decision boundary must classify all the points correctly

SVM-formula-Example-1

* The decision boundary can be ascertained by solving the subsequent constrained optimization problem



* As this is a constrained optimization problem. Solving it requires Lagrange multipliers

SVM-formula-Example-3

* The Lagrangian is

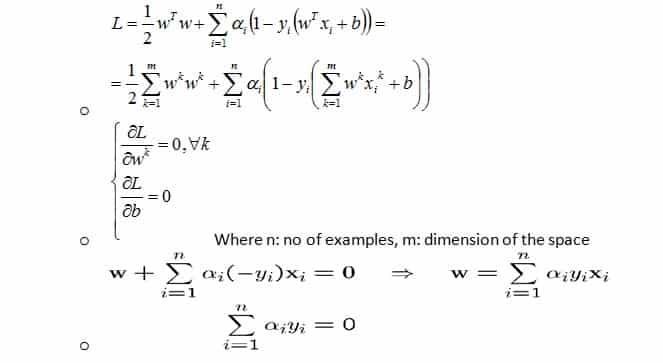
SVM-formula-Example-4

ai≥0

– Note that ||w||² = wΤw

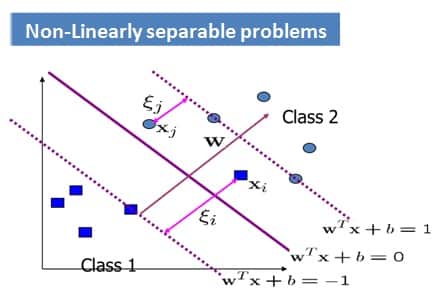
* Gradient with respect to w and b

Setting the gradient of L w.r.t. w and b to zero, we have

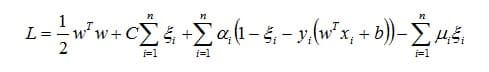


However for Non-Linearly separable problems

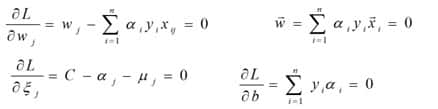
* We allow the “error” xi in classification, it is based on the output of the discriminant function wTx+bo
* xi approximates the number of misclassified samples



Now the Optimization Problem becomes:



With α and μ Lagrange multipliers, POSITIVE



Extending the same to a Non-Linear margin or decision boundary

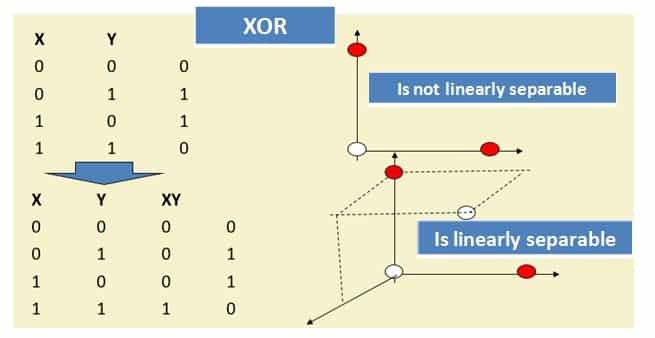
We need to transform xi to a higher dimensional space to make our life easier

* I**nput space:** the space where the point xi are located
* **Feature space:** the space of f(xi) after transformation

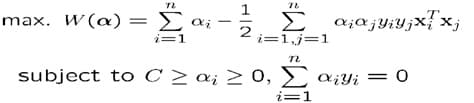
## **hy do we need to transform?**

* Linear operation present in the feature space is equivalent to non-linear operation in the input space
* Classification can become easier with a proper transformation.

This can be illustrated with an XOR problem, where adding a new feature of x1x2 makes the problem linearly separable.



So now as per SVM optimization problem,



The data points appear only as inner product (Xi Xj).

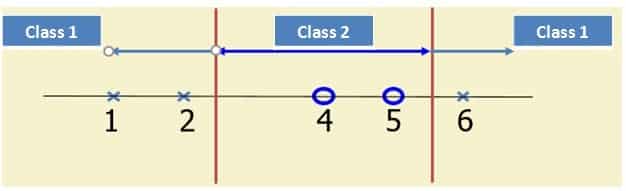
As long as we can compute the inner product in the feature space, we do not require the mapping explicitly. **This is called Kernel Trick.**

Several common and known geometric operations (angles, distances) can be articulated by inner products. We now can define the kernel function K by

SVM-formula-Example-8

The Kernel Trick can be illustrated with an example:

* Suppose we have 5 one Dimensional data points with 1, 2, 6 as class 1 and 4, 5 as class 2



We use the 2nd degree polynomial kernel

* K(x,y) = (xy+1)2
* C is set to 100

SVM-formula-Example-9

SVM-formula-Example-10

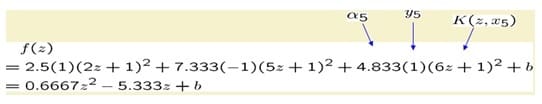
* We first find ai (i=1, …, 5)
* By using a Quadratic problem solver, we get

a1=0, a2=2.5, a3=0, a4=7.333, a5=4.833

Note that the constraints are indeed satisfied

The support vectors are {x2=2, x4=5, x5=6}

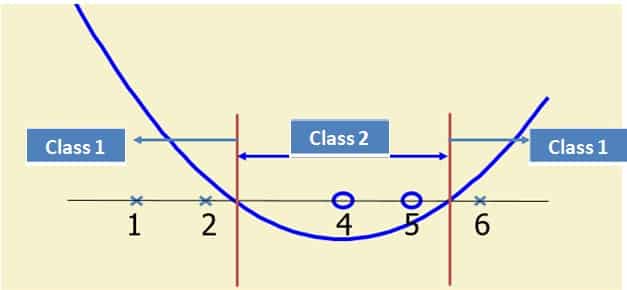
* The discriminant function is



b is found by solving f(2)=1 or f(5)=-1 or by f(6)=1,

All three would give b=9

SVM-formula-Example-13



Few Popular Kernels: The most tricky and demanding part of using SVM is to choose the right Kernel function because it’s very challenging to visualize the data in n-dimensional space. Few popular kernels are:

* **Fisher Kernel:** It is a kernel function that analyses and measures the similarity of two objects. This is done on the basis of sets of measurements for each object and a statistical model.
* **Graph Kernel:** It is a kernel function that computes an inner product on graphs.
* **Polynomial Kernel:** It is a kernel commonly used with support vector machines (SVMs). It is also used with other kernelised models that symbolizes the similarity of vectors in a feature space over polynomials of the original variables, allowing learning of non-linear models.
* **Radial Basis Function Kernel (RBF):** It is a real-valued kernel function whose value depends only on the distance from the origin, or distance from some other point called a centre.