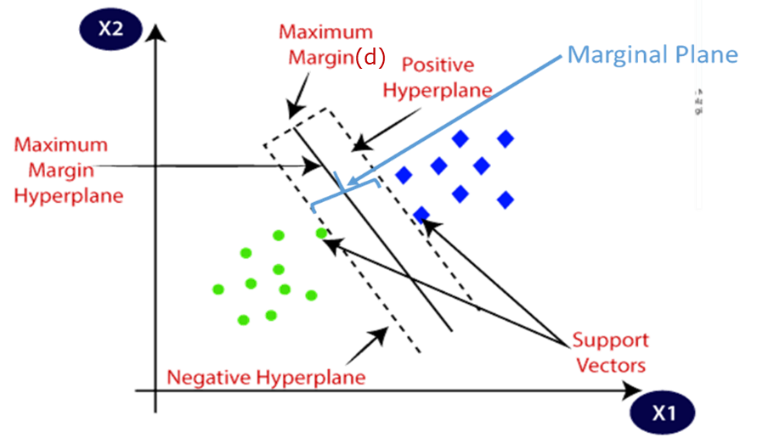
**Support Vector Machine:**

Support Vector Machine is used for Classification(SVC) as well as Regression(SVR) problems. However, primarily, it is used for Classification problems.

The primary objective of an SVM is to find the **optimal hyperplane** that best **divides the data points of different classes while maximizing the margin between the two classes**. The margin is defined as the distance between the hyperplane and the nearest data points of each class (known as support vectors). SVM aims to find the hyperplane that not only separates the classes but also generalizes well, to new unseen data.



**Support Vectors:** Support vectors are **data points that are closer** to the **hyperplane** and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. These are the points that help us build our SVM.

**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 the image), the n hyperplane will be a straight line. And if there are 3 features, then the n hyperplane will be a 3-dimensional plane.

**Positive Hyperplane:** The boundary line(Hyperplane) situated at the positive region is known as the Positive Hyperplane

**Negative Hyperplane:** A boundary line situated at a negative region is known as Negative Hyperplane.

**Marginal plane OR Maximum margin (d):** the distance between two hyperplanes (Positive and negative Hyperplane).

**Hard Margin:** A margin clearly separates the data points by using a marginal plane with no error.

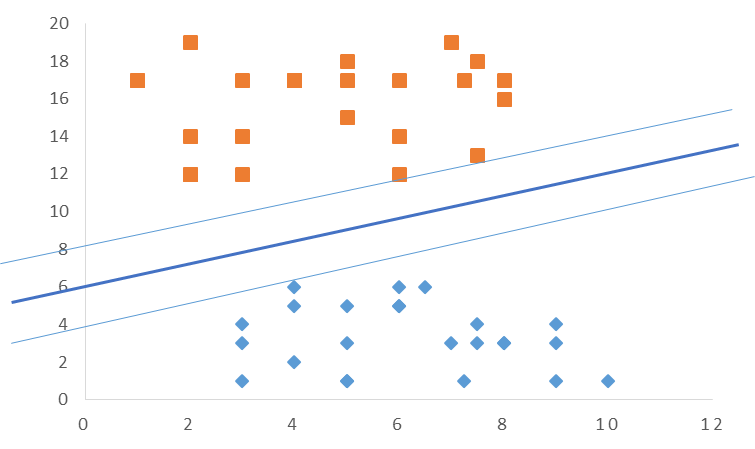
**Soft Margin:** Here marginal plane is not clearly separating the data points and you will observe some errors.

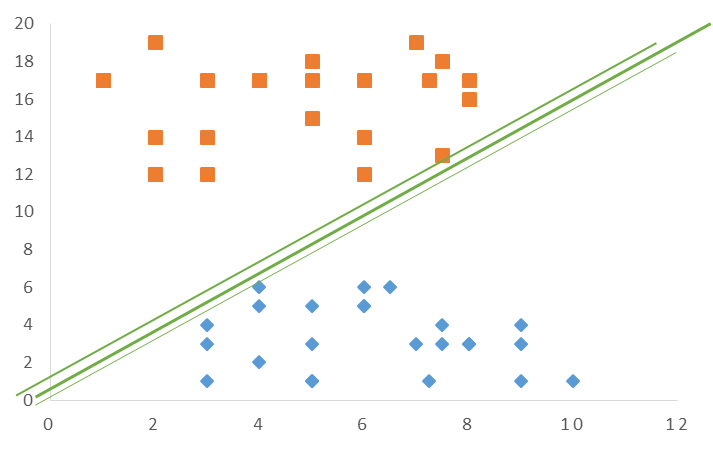
**Support vector Classifier:**

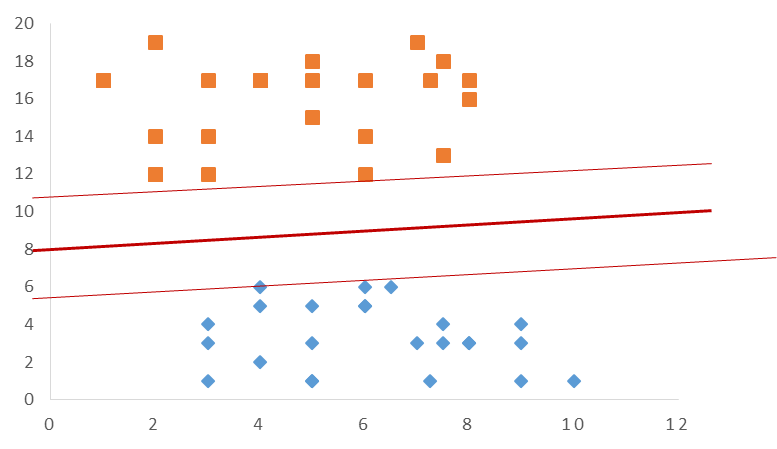
**Geometrical intuition for SVC:**

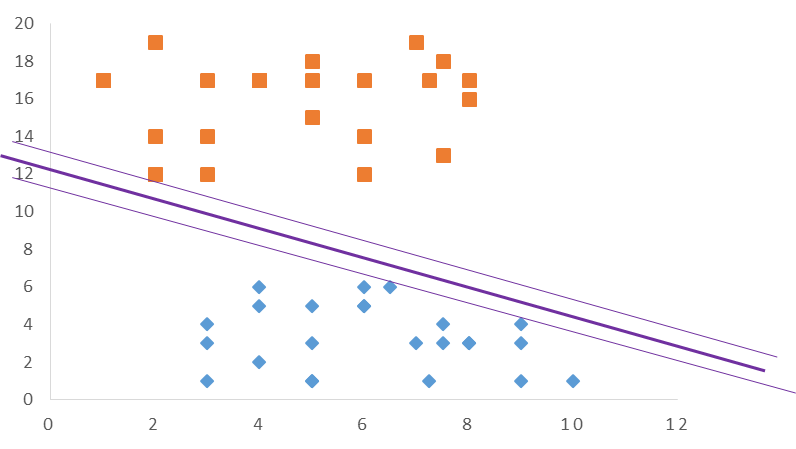
1. Create the hyperplane(line) which divides the points.
2. Find the points that falls on the +ve side and -ve side of the Hyperplane
3. Then find the +ve support vector and –ve Support vector
4. Basis of the support vector draw +ve Hyperplane and -ve Hyperplane.
5. Repeat the process until you find the maximum d (distance between +ve hyperplane and –ve hyperplane)

Here we have repeated the process for 4 times, please observe the below images. You will observe that the 4th image has the maximum distance

****

****

****

****

**Mathematical intuition for SVC:**

The distance between the marginal plane should be high. The same can be achieved by changing values of w, b.

1. Create a straight line for two-dimensional and Hyperplane for n-dimensional data which should divide the points.
2. Find the points that fall on the +ve side of the hyperplane and determine the support vectors then using support vectors create a Positive Hyperplane.
3. Then Find the points that fall on the -ve side of the hyperplane and determine the support vectors then using the support vector create a Negative Hyperplane.
4. Find the data falling within the support vector.
5. **Create a straight line for two-dimensional and Hyperplane for n-dimensional data which should divide the points.**

|  |
| --- |
| 2 Dimension |
| * In two-dimensional data a straight line will be created which will separate the data. |

|  |
| --- |
| 3 Dimension |
| * In “three” or “n” dimension data a hyperplane will be created which will separate the data as shown in the above image. |

**The equation for a n Dimension that is more than three.**

For n-dimensions, the equation can be expressed as follows.

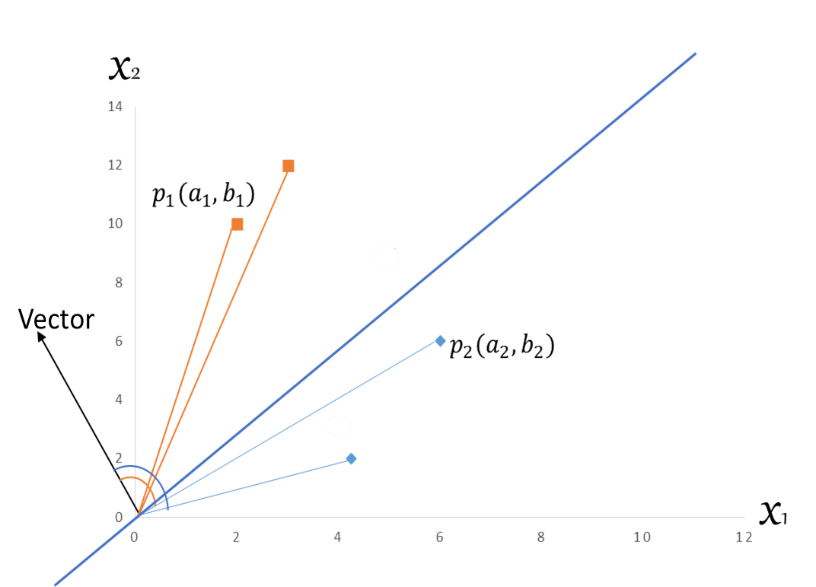
To handle the majority of cases that involve n-dimensional data, it can be challenging to apply the existing equation directly. To improve usability, we can simplify the equation, as follows.

For better clarity in understanding SVC, we can assume that the hyperplane passes through the origin. With this assumption, hence if the hyperplane passes through the origin the value of “c” will be 0, resulting in the equation becoming as follows.

The above equation is a generalized form that can be used for data in any dimensionality, including two-dimensional, three-dimensional, and even n-dimensional data. It offers flexibility and applicability across different dimensionalities.

By using the above equation, we can create the **hyperplane in SVM.**

1. **Find the points that fall on the +ve side of the hyperplane and determine the support vectors then using support vectors create a Positive Hyperplane.**
2. **Then Find the points that fall on the -ve side of the hyperplane and determine the support vectors then using the support vector create a Negative Hyperplane.**

****

The equation represents the hyperplane. However, this equation alone cannot be used to determine the positions of points falling above or below the hyperplane. To achieve that, we need to make some modifications to the equation.

To locate points with respect to the hyperplane, we divide the equation by , where represents the magnitude of the weight vector. The modified equation is as follows:

|  |
| --- |
| The above equation can be further simplified as below. |

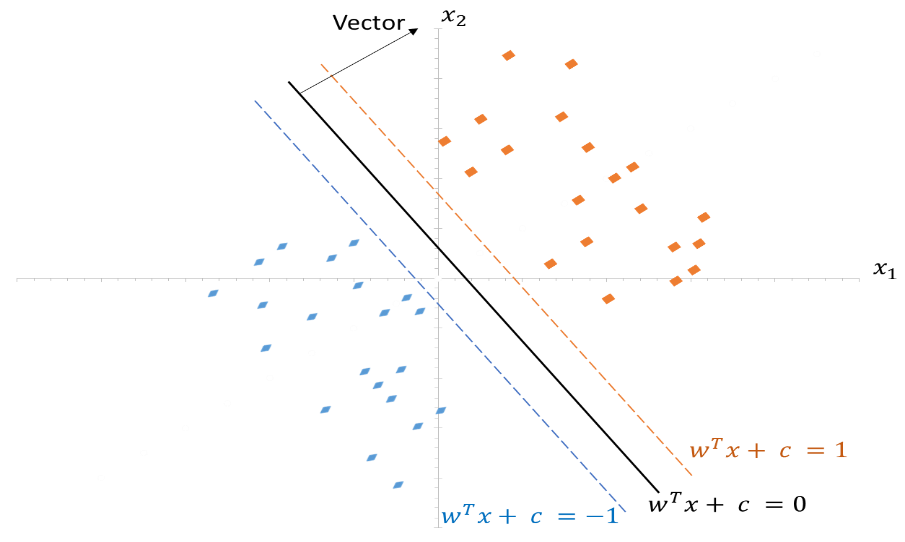
Using the above equation, we can determine the position of the points (values) relative to the hyperplane, weather it lies above or below the hyperplane.

The crucial factor in this equation is the role played by cosine theta (cosθ), as it helps ascertain the point's position based on the angle it forms with the vector.

Points above the hyperplane exhibit an angle less than 90 degrees, while points below the hyperplane have an angle greater than 90 degrees (as depicted in the figure). Therefore, when applying the cosθ value in the equation,   
- points with an angle **less** than 90 degrees yield a **positive value**,   
- while those with an angle **greater** than 90 degrees result in a **negative value**.

Consequently, we can conclude that all positive values lie above the plane, whereas all negative values lie below the hyperplane.

1. **Create the Marginal Plane(+ve and –ve Hyperplane)**

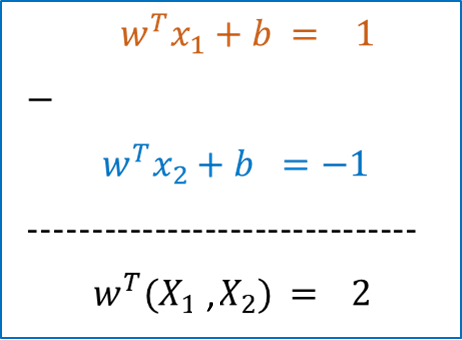


Dotted orange and dotted blue lines are the support vectors created by using the equation as shown in the figure

Once the Marginal Plane is created we need to ensure that the distance between the +ve and –ve Hyperplane should be maximum.

however, the question here is How do we maximize the distance? And how to do we achieve it. The same can be achieved by following the below steps.

1. Subtract the equation of Hyperplane.
2. Obtain the unit vector
3. Enhance the accuracy
4. **Subtract the equation of support vectors.**



1. **Obtain the unit vector:**

The unit vector can be obtained by dividing the above equation by . (For hard margin)

1. **Enhance accuracy:**

It is crucial to increase the distance between the hyperplanes, which is measured by the expression . In other words, maximizing is actually increasing the distance between two planes and it is also necessary for improving accuracy.

Achieving this goal involves adjusting the values of W and b, which determine the hyperplane's orientation and position. By finding the optimal values for W and b through methods like gradient descent or other optimization algorithms, we can increase the magnitude of W (represented as within the absolute value function, resulting in a larger denominator and thus a higher value for .

Consequently, by modifying the values of W and b, we can effectively increase the distance between the hyperplanes and enhance accuracy.

How to identify all the correct classified points?

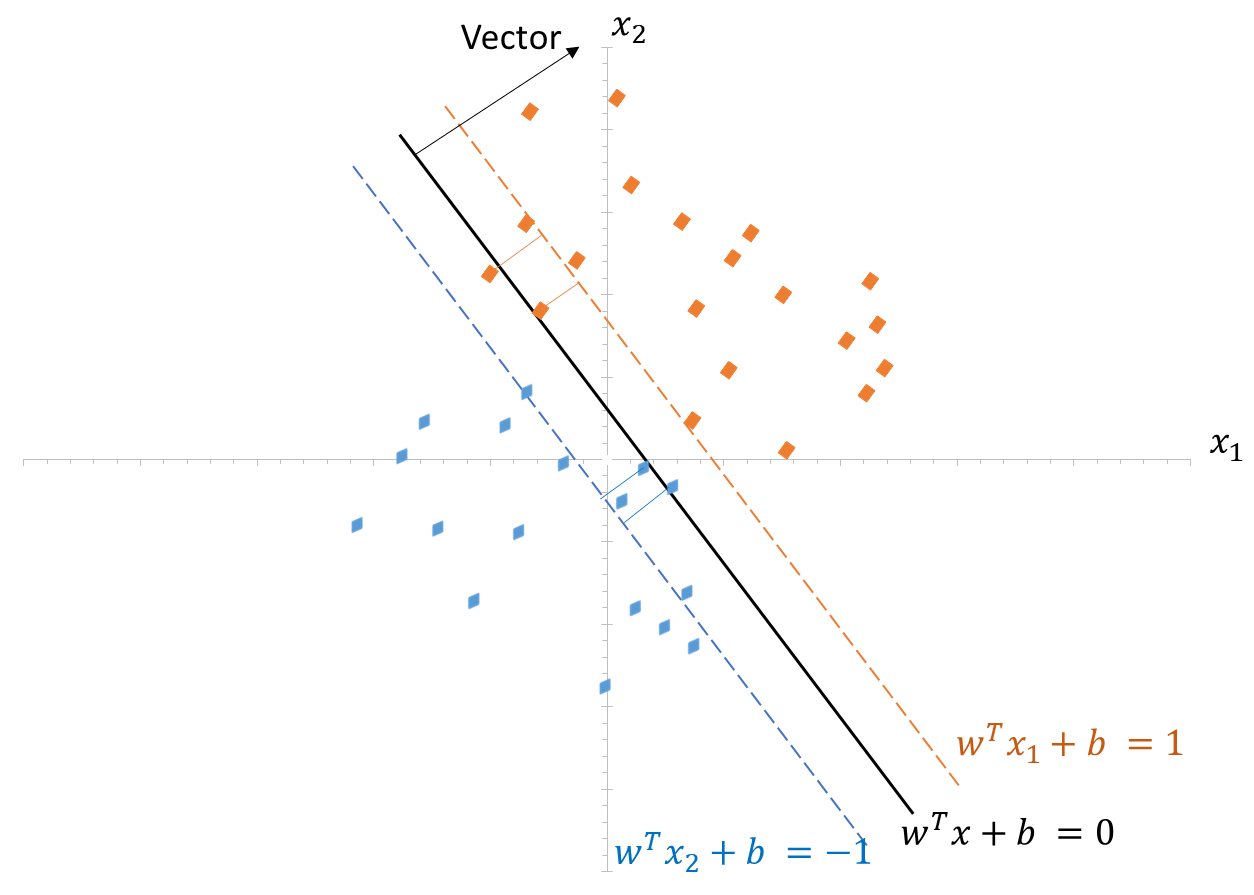
|  |  |  |
| --- | --- | --- |
|  |  | Correct predicted output |
|  | If is greater than or equal to 1  then the correct predicted  output is |  |
| If is greater than or equal to -1  then the correct predicted  output is |  |

Hence we can prove that if we multiply with the it will return a value that is greater than or equal to **+1**. Hence more simplified equation can be written below

The equation i.e. is commonly applied when the margin(Hard Margin) clearly separates the data points without any errors.

**Process for Soft Margin**

However, in most cases, there will be some errors present within the marginal plane, as illustrated in the image below. Consequently, utilizing the above equation alone will not yield an accurate model. To address such data scenarios, modifications are necessary for the equation



To handle data with errors in the marginal plane, adjustments should be made to the equation. This can involve reducing the marginal plane distance and incorporating hyper-parameter techniques,

1. reducing the marginal plane can be achieved by using , Instead of .
2. Here hyper parameter techniques, such as addition and regularization techniques such as the addition of can be employed to penalize to reduce excessive incorrect classifications.

In summary, to tackle data with errors in the marginal plane and create a more accurate model, modifications to the equation, such as incorporating regularization techniques, are essential. This enables the model to strike a balance between maximizing the margin and reducing errors, resulting in improved performance.

The updated equation is as below.

Sum of the distance of all incorrectly classified points. Here the distance will be taken from the marginal plane.

**Hinge Loss:** In machine learning, hinge loss is a loss function used for training classifiers. The hinge loss is used for "maximum margin" in support vector classification,

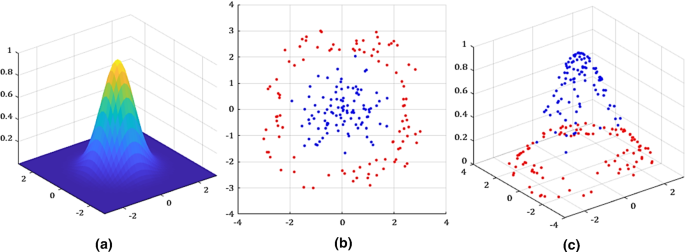
**What is SVM Kernal?**

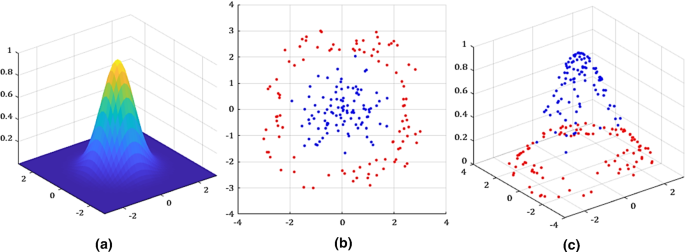
This allows SVM to effectively handle data that cannot be separated by a straight line.

In short, the SVM kernel is a mathematical function that allows SVM to handle complex, **non-linear data** by implicitly transforming it into a higher-dimensional space. It enables SVM to find an **optimal hyperplane** that can effectively separate different classes in the transformed space, even when they **are not linearly separable** in the original feature space. Popular kernels include,

1. **Polynomial,**
2. **Radial basis function (RBF),**
3. **Sigmoid kernels,**

Each is suited for different types of data patterns. The choice of the kernel is crucial for SVM's performance, and it plays a key role in extending SVM's application to various classification problems.





**Polynomial Kernel:** The polynomial kernel computes the dot product after raising it to a certain power "d", making it suitable for data that can be separated by polynomial decision boundaries.

**K(x, y) = (x · y + c)^d**

**Radial Basis Function (RBF) Kernel:** The RBF kernel, also known as the Gaussian kernel, is commonly used for non-linearly separable data. It transforms the data into an infinite-dimensional feature space.

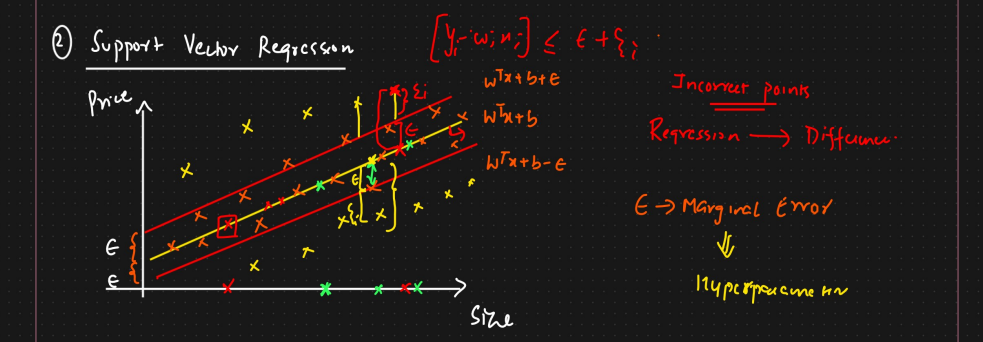
K(x, y) = exp(-γ \* ||x - y||^2)

Here, "γ" is a hyperparameter that controls the smoothness of the decision boundary.

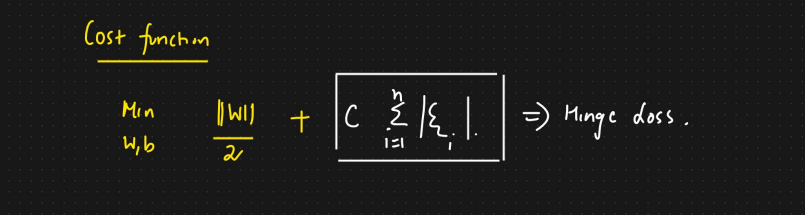
**Sigmoid Kernel:** The sigmoid kernel is another option, but it is less commonly used. It is useful when dealing with neural networks and binary classification problems.

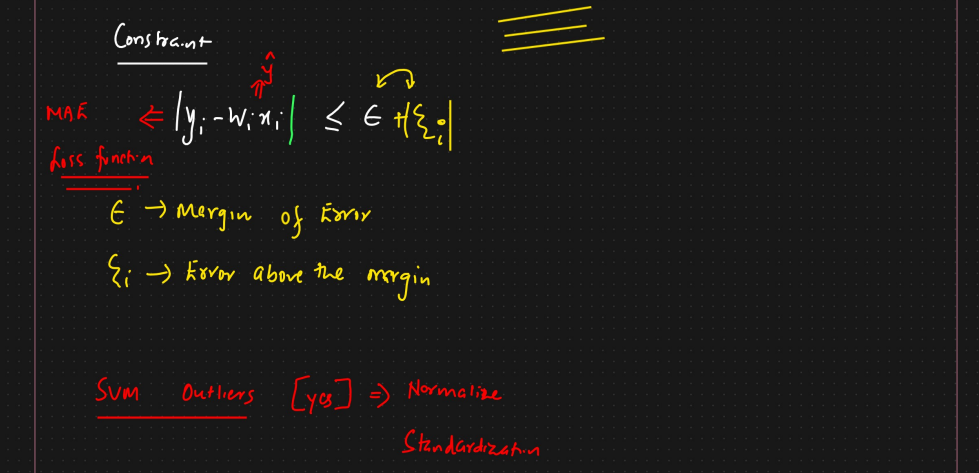
K(x, y) = tanh(α \* x · y + c)

**Support Vector Regressor:**

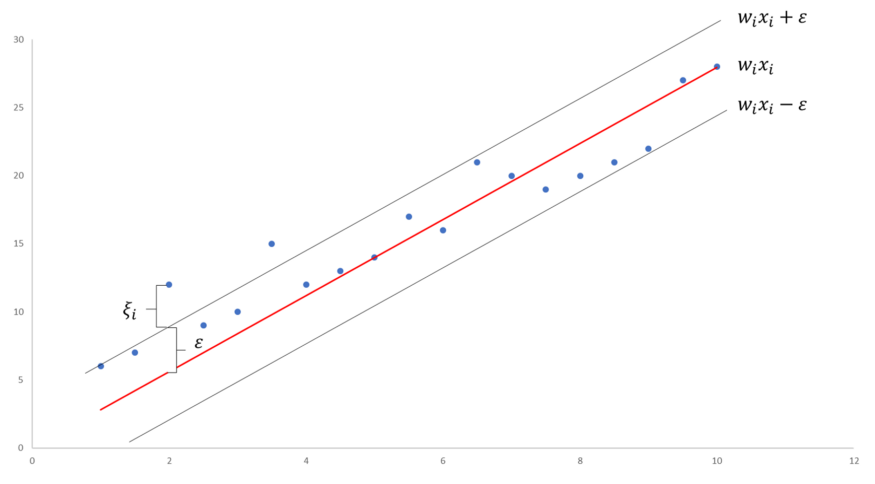


In support vector regressor some amount of epsilon(ε)(Marginal Error) is ok .





In support vector regressor some amount of epsilon(ε)(Marginal Error) + ζi is ok.



Svm KERNAL:

1. Polynomial kernal
2. RGB kernal
3. Sigmoid kernal