Support Vector Machine

1. Introduction of SVM
2. What is Hyperplane, Support Vector and Margin
3. How to Select Optimal Hyperplane
4. Soft Margin and Hard Margin SVM
5. Regulation in SVM
6. What is Kernel
7. Types of Kernel
8. Tuning Parameter: Regulation, gamma

Support vector machines (**SVMs**) **are supervised learning models with associated learning algorithms** that analyze data used for classification and regression analysis. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane (MMH) that best divides the dataset into classes.

SVM approach has some advantages compared to others classifiers. They are robust, accurate and very effective even in cases where the number of training samples is small. SVM technique also shows greater ability to generalize and greater likelihood of generating good classifiers.

Wikipedia tells us that SVMs can be used to do two things: classification or regression.

1. SVM is used for classification
2. SVR (Support Vector Regression) is used for regression

Goal of SVM:

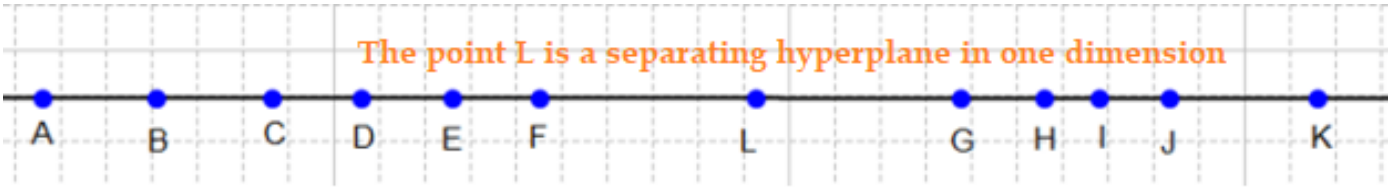
The goal of a support vector machine is to find the optimal separating hyperplane which maximizes the margin of the training data.

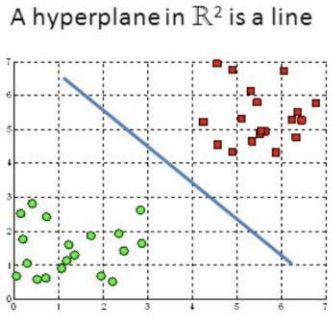
Hyperplane

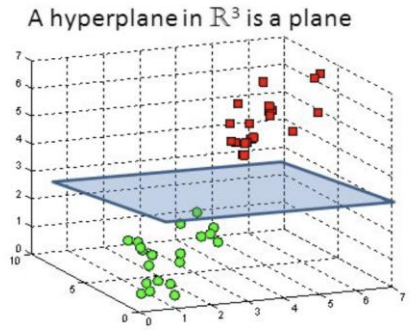
A hyperplane is a decision plane which separates between a set of objects having different class memberships.

A hyperplane is a generalization of a plane.

- in one dimension, a hyperplane is called a point
- in two dimensions, it is a line
- in three dimensions, it is a plane
- in more dimensions you can call it a hyperplane





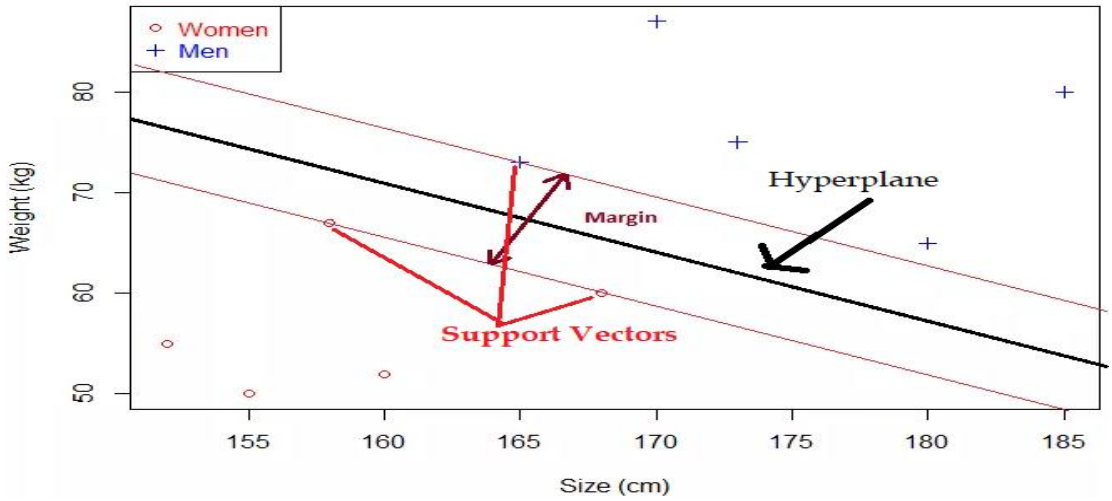


Support Vectors

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins.

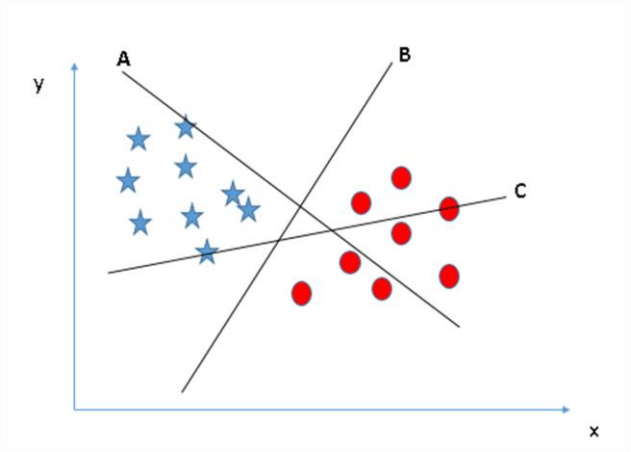
Margin

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

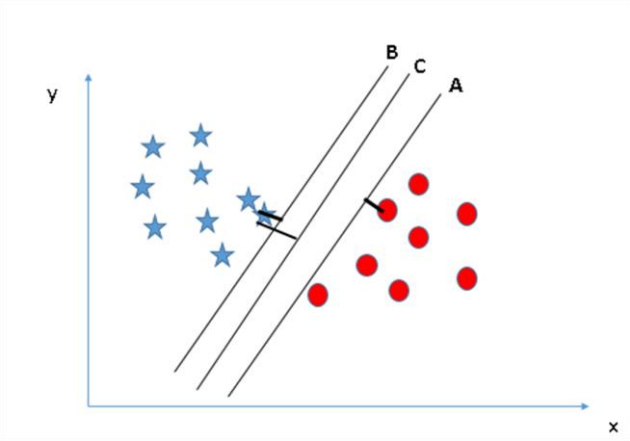


How can we identify the right hyper-plane?

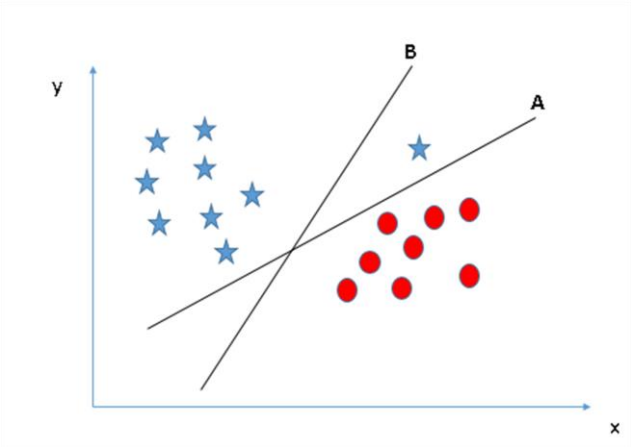
1. The two data points should be separated by maximum distance.
2. These two data points closer to the margin
3. Generate hyperplanes which segregates the classes in the best way



Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job:- Rule 3



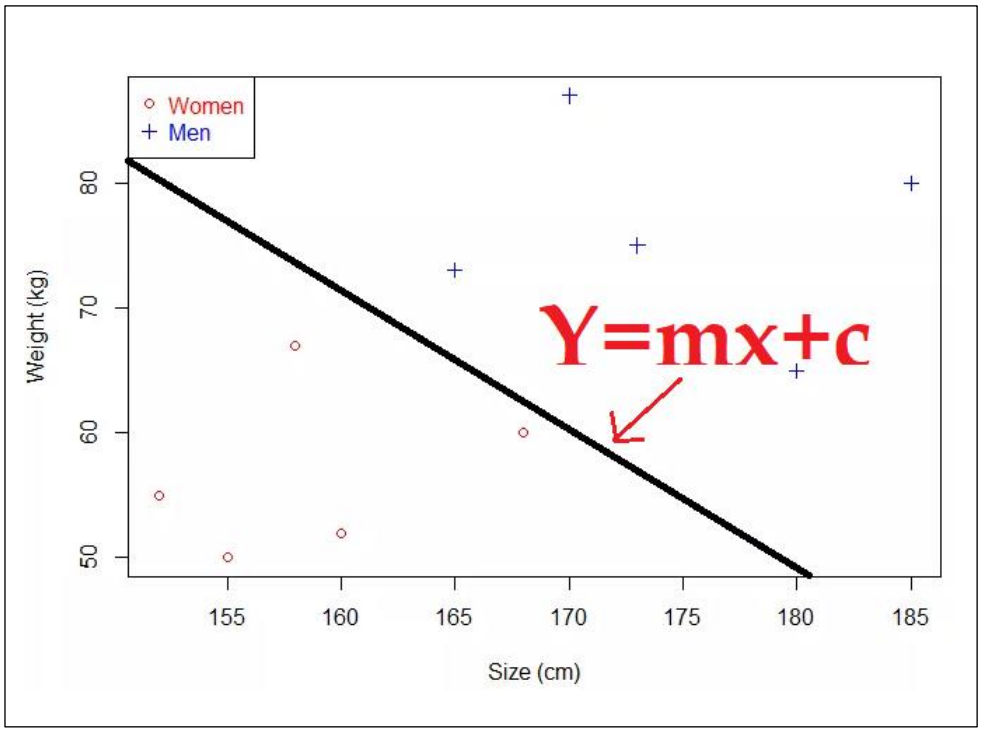
We can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C.:- Rule 1,2



SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

How to Select Optimal Hyperplane?

Understanding the equation of the hyperplane



Equation of line:

$$y = mx + b$$

$$y - mx - b = 0 \quad \text{.....(Eq 1)}$$

Given two vector W and X in 3-dimension

$$W = \begin{bmatrix} -b \\ -m \\ 1 \end{bmatrix} \quad and \quad X = \begin{bmatrix} 1 \\ x \\ y \end{bmatrix}$$

$$W^T X = [-b - m \ 1] \begin{bmatrix} 1 \\ x \\ y \end{bmatrix}$$

$$= (-b \times 1) + (-m \times x) + (1 \times y)$$

$$= y - mx - b \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (Eq\ 2)$$

Substitute Eq 2 in Eq 1 then equation of Hyperplane is,

$$\boldsymbol{W^T X = 0}$$

Why do we use the hyperplane equation $W^T X$ **Instead of** $y = mx + b$**?**

1. It is easier to work in more than two dimensions with this notation
2. The vector w will always be normal to the hyperplane

Convert 3-dimension to 2-dimension

Given, W= [-m 1] and X= [x y]

$$W.X = (-m \times x) + (1 \times y)$$

$$= -mx + y$$

$$= W^T X + b$$

$$W.X - b = W^T X$$

Equation of Hyperplane in 2-Dimension:

$$\boldsymbol{W.X - b = 0}$$

How can we find the biggest margin?

1. Select two hyperplanes which separate the data with no points between them
2. Maximize their distance (the margin)

We can select two others hyperplanes H1 and H2 which also separate the data and have the following equations:

$$\boldsymbol{W.X - b = \delta}$$

$$\boldsymbol{W.X - b = -\delta}$$

If $\delta = 1\ then,$

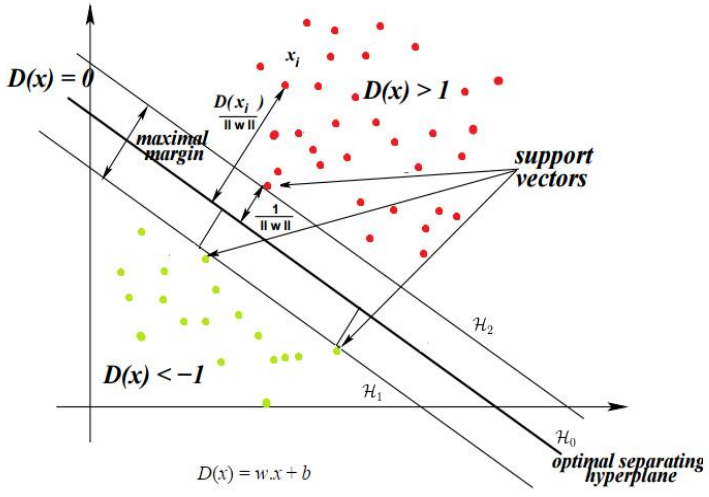
$$\boldsymbol{W.X - b = 1}$$

$$\boldsymbol{W.X - b = -1}$$

For each vector xi either:

$W.X_i - b \geq 1 \quad for\ X_i\ having\ class\ 1$

$W.X_i - b \leq -1 \quad for\ X_i\ having\ class\ -1$



Combining the constraints

Both the constraints stated above can be combined into a single constraint.

Constraint 1:

For xi having the class -1, $(w.x - b) \leq 1$

Multiplying both sides by y_i (which is always -1 for this equation)

$y_i(w.x - b) \geq y_i(-1)$ which implies $y_i(w.x - b) \geq 1$ for xi having the class?1.

Constraint 2:

$y_i(w.x - b) \geq 1$ for xi having the class 1

Combining both the above equations, we get

$$\boldsymbol{y_i(w.x - b) \geq 1 \; for \; 1 \leq i \leq n}$$

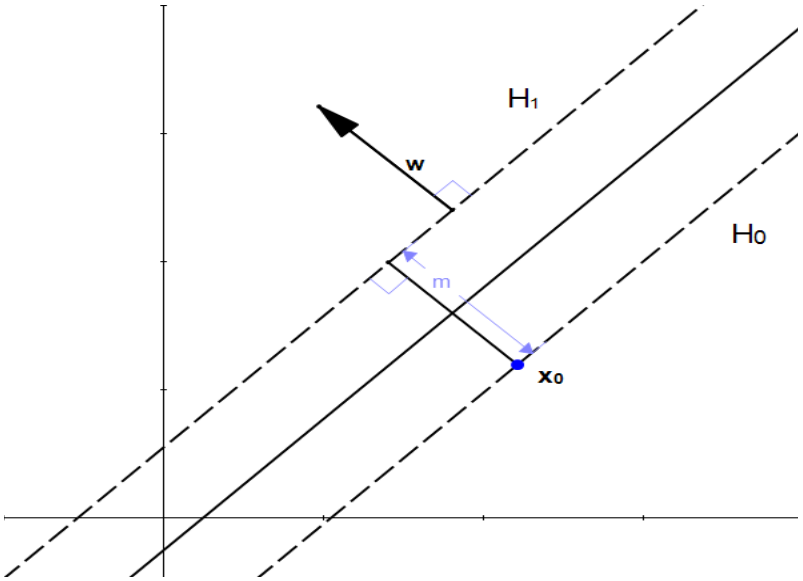
This leads to a unique constraint instead of two which are mathematically equivalent. The combined new constraint also has the same effect, i.e., no points between the two hyperplanes.

Hard margin SVM

If $\boldsymbol{y_i(w.x - b) \geq 1}$
then Xi is correctly classified
else:
Xi is incorrectly classified.

we can see that if the points are linearly separable then only our hyperplane is able to distinguish between them and if any outlier is introduced then it is not able to separate them. So, these type of SVM is called *as* ***hard margin SVM***.

Step 2: Maximize the distance between the two hyperplanes



Let:

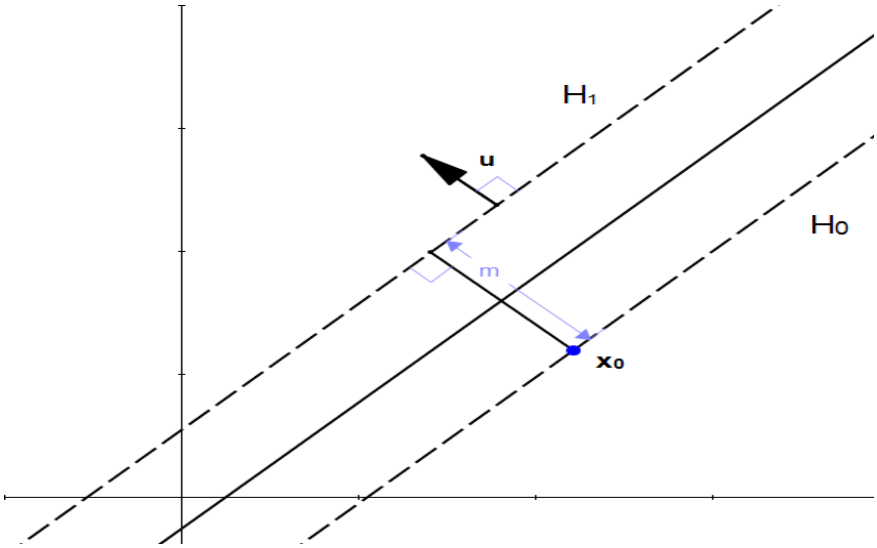
- H0 be the hyperplane having the equation $(\boldsymbol{w}.\boldsymbol{x} - \boldsymbol{b}) = \mathbf{-1}$
- H1 be the hyperplane having the equation $(\boldsymbol{w}.\boldsymbol{x} - \boldsymbol{b}) = \mathbf{1}$
- x0 be a point in the hyperplane H0.

We will call m the perpendicular distance from x0 to the hyperplane H1. By definition, m is what we are used to call **the margin**.

As x0 is in H0 , m is the distance between hyperplanes H0 and H1.

we already know a vector perpendicular to H1, that is w (because H1=$(\boldsymbol{w}.\boldsymbol{x} - \boldsymbol{b}) = \mathbf{1}$)

Let's define $u = w/||w||$ the unit vector of w. As it is a unit vector ‖u‖=1 and it has the same direction as w so it is also perpendicular to the hyperplane.

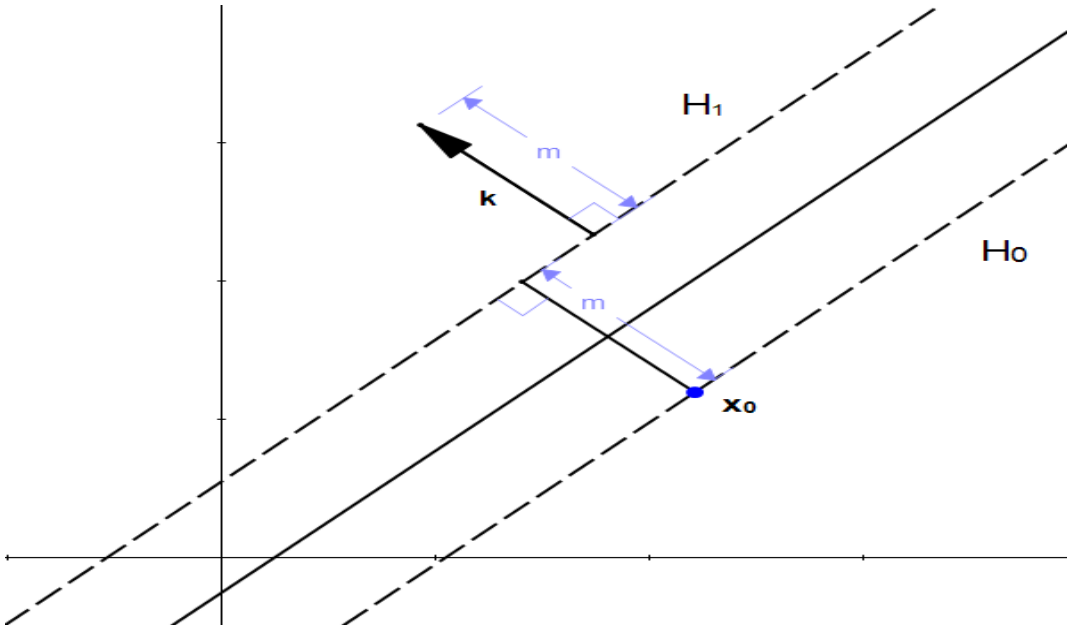


If we multiply u by m we get the vector k=mu and :

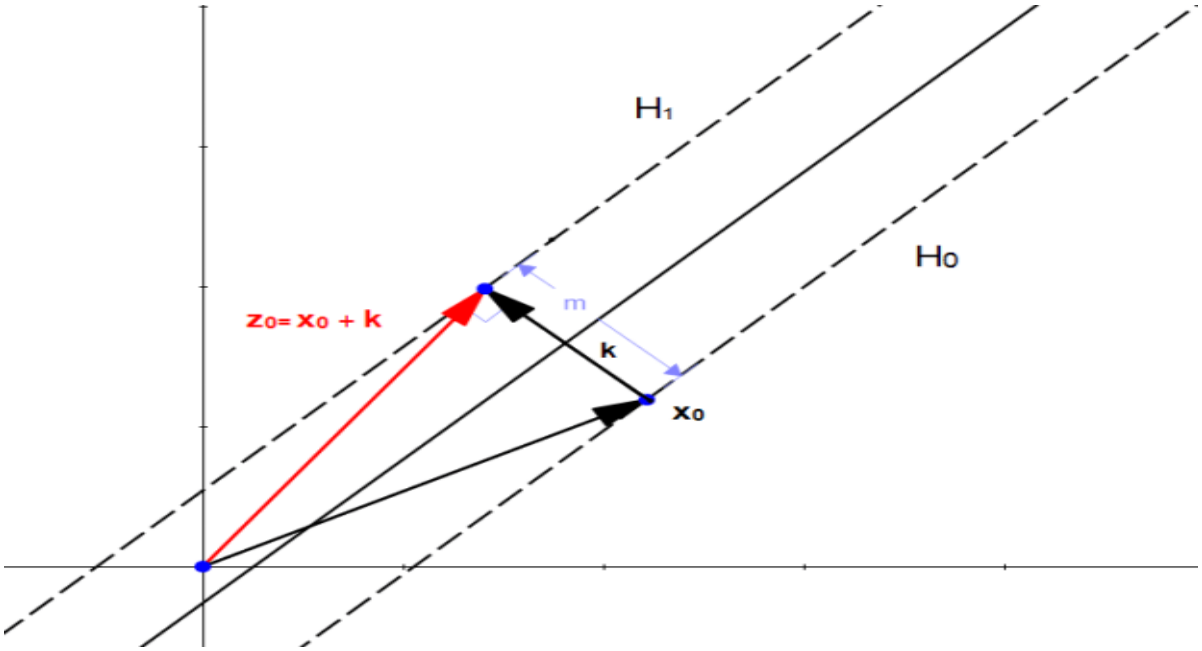
1. ‖k‖=m
2. k is perpendicular to H1(because it has the same direction as u)

From these properties we can see that k is a vector of length m perpendicular *to H1*

$$K = mu = m\frac{W}{||W||}$$



If we start from the point X_0 and add k we find that the point $z_0 = X_0$+K is in the hyperplane H1



The fact that z0 is in H1 means that

$$W.Z_0 - b = 1$$

$$W.(x_0 + K) - b = 1$$

$$W.\left(x_0 + m\frac{W}{||W||}\right) - b = 1$$

$$W.x_0 + m\frac{||W||^2}{||W||} - b = 1$$

$$W.x_0 - b = 1 - m||W||$$

As x0 is in H0 then w·x0-b=−1

$$-1 = -1 - m||W||$$

$$\boldsymbol{m = \frac{2}{||W||}}$$

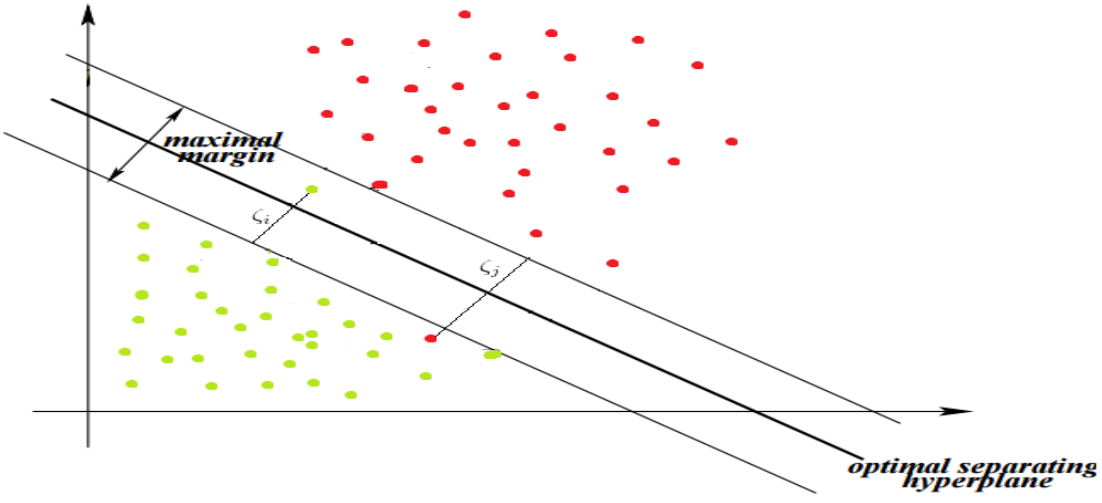
Our goal is to maximize the margin. Among all possible hyperplanes meeting the constraints, we will choose the hyperplane with the smallest ‖w‖ because it is the one which will have the biggest margin.

The only variable in this formula is w, which is indirectly proportional to m, hence to maximize the margin we will have to minimize ||w||. This leads to the following optimization problem:

$$\text{Minimize in } (w, b)\ \frac{||W||^2}{2} \text{ subject to } y_i(w.x - b) \geq 1 \text{ for any } i = 1, \ldots, n$$

The above is the case when our data is linearly separable. There are many cases where the data cannot be perfectly classified through linear separation. In such cases, Support Vector Machine looks for the hyperplane that maximizes the margin and minimizes the misclassifications.

For this, we introduce the slack variable,$\varepsilon_i$which allows some objects to fall off the margin but it penalizes them.



In this scenario, the algorithm tries to maintain the slack variable to zero while maximizing the margin. However, it minimizes the sum of distances of the misclassification from the margin hyperplanes and not the number of misclassifications.

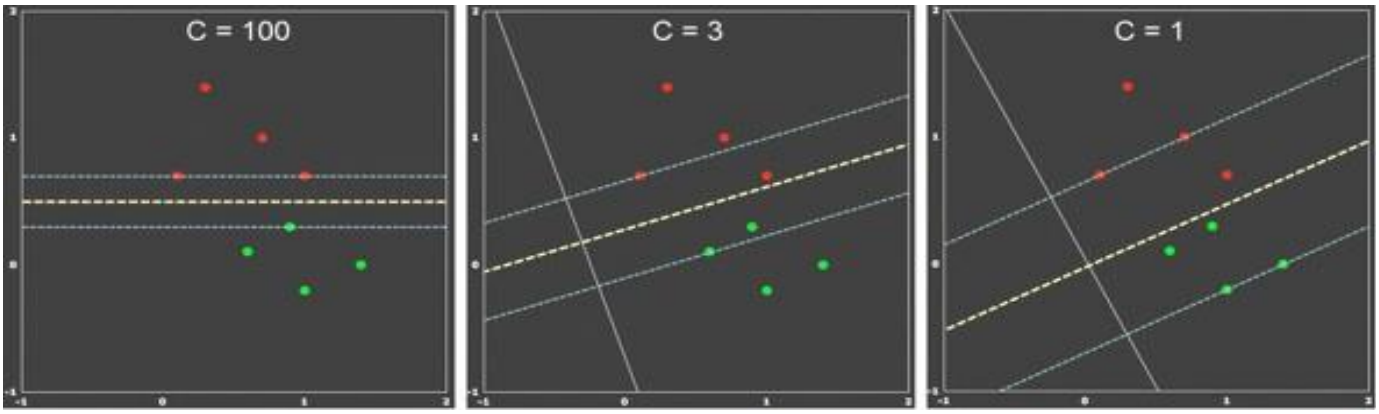
Constraints now changes to $\boldsymbol{y_i\,(w.x-b)\geq 1-\varepsilon_i\; for\; 1\leq i<n, \varepsilon_i\;\geq 0}$ and the optimization problem changes to

$$Minimize\ in\ (w,b)\frac{\left||w|\right|^2}{2}+C\sum_i \varepsilon_i\ subject\ to y_i\ (w.x-b)\geq 1-\varepsilon_i\ for\ any\ i=1\ldots..n$$

Here, the parameter ***C*** is the ***regularization*** parameter that controls the trade-off between the slack variable penalty (misclassifications) and width of the margin.

1. Small C makes the constraints easy to ignore which leads to a large margin.
2. Large C allows the constraints hard to be ignored which leads to a small margin.
3. For C=∞, all the constraints are enforced.

Large Value of parameter C => small margin
Small Value of parameter C => Large margin



Soft margin SVM:

We basically consider that the data is linearly separable and this might not be the case in real life scenario. We need an update so that our function may skip few outliers and be able to classify almost linearly separable points. For this reason, we introduce a new *Slack variable ($\boldsymbol{\varepsilon}$)* which is called X_i.

if we introduce $\boldsymbol{\varepsilon}$ it into our previous equation we can rewrite it as

$$\boldsymbol{y_i\,(w.x-b)\geq 1-\varepsilon_i}$$

if $\boldsymbol{\varepsilon_i}$ = 0,
the points can be considered as correctly classified.
else:
$\boldsymbol{\varepsilon_i}$> 0, Incorrectly classified points.

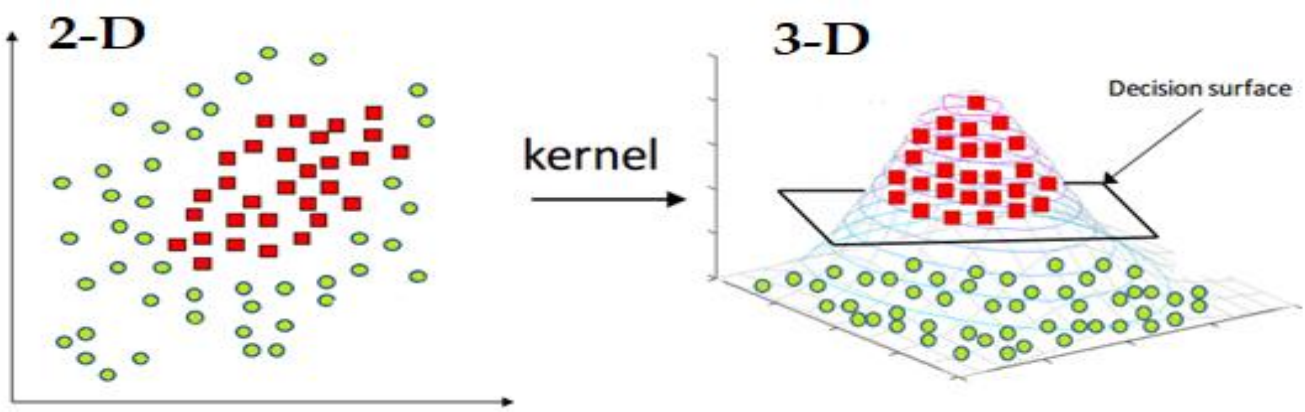
so, if $\boldsymbol{\varepsilon_i}$> 0 it means that X_i (variables)lies in incorrect dimension, thus we can think of ξi as an error term associated with X_i (variable). The average error can be given as;

$$\frac{1}{n}\sum_i^n \varepsilon_i$$

Kernel:

The easiest way to separate two classes of data is a line in case of 2D data and a plane in case of 3D data. But it is not always possible to use lines or planes and one requires a nonlinear region to separate these classes. Support Vector Machines handle such situations by using a kernel function which maps the data to a different space where a linear hyperplane can be used to separate classes. This is known as the **kernel trick** where the kernel function transforms the data into the higher dimensional feature space so that a linear separation is possible.

The kernel takes a low-dimensional input space and transforms it into a higher dimensional space. In other words, we can say that it converts no separable problem to separable problems by adding more dimension to it. It is most useful in non-linear separation problem. Kernel trick helps you to build a more accurate classifier.



1. **Linear Kernel:** A linear kernel can be used as normal dot product any two given observations. The product between two vectors is the sum of the multiplication of each pair of input values.

$$K(X, X_i) = Sum(X * X_i)$$

The kernel defines the similarity or a distance measure between new data and the support vectors. The dot product is the similarity measure used for linear SVM or a linear kernel because the distance is a linear combination of the inputs.

2. **Polynomial Kernel:** A polynomial kernel is a more generalized form of the linear kernel. The polynomial kernel can distinguish curved or nonlinear input space.

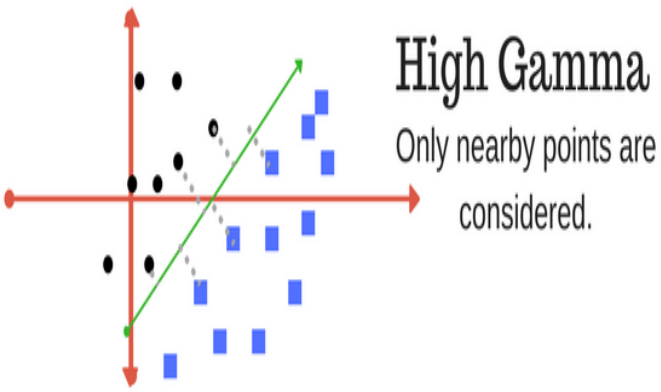
$$K(X, X_i) = 1 + Sum(X * X_i)^d$$

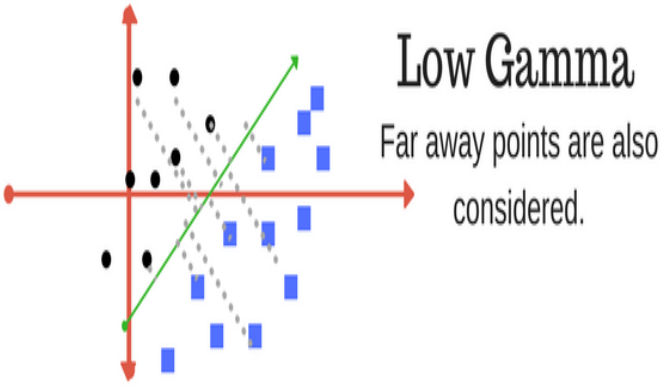
Where d is the degree of the polynomial. d=1 is similar to the linear transformation. The degree needs to be manually specified in the learning algorithm.

3. **Gaussian Radial Basis Function Kernel (*RBF*):** The Radial basis function kernel is a popular kernel function commonly used in support vector machine classification. RBF can map an input space in infinite dimensional space.

$$K(X, X_i) = \exp(-\text{gamma} * Sum(X - X_i^2))$$

Here gamma (γ) is a parameter, which ranges from 0 to 1. A higher value of gamma will perfectly fit the training dataset, which causes over-fitting. Gamma=0.1 is considered to be a good default value. The value of gamma needs to be manually specified in the learning algorithm.





4. **Sigmoid Kernel***:* Sigmoid kernel is also known as hyperbolic tangent (sigmoid) kernel and multilayer perception (MLP) kernel. The origin of this kernel is from theory of neural networks and has been found to be performed well in practice as well. There are two adjustable parameters of sigmoid kernel, ***slope of alpha*** and the ***intercept parameter C***. Scoring function of sigmoid kernel is as follow:

$$K(X, X_i) = \tanh(\alpha x x^T + c)$$

Tuning Parameters:

1. **C**: Inverse of the strength of regularization.
   Behaviour: As the value of 'c' increases the model gets overfits.
   As the value of 'c' decreases the model underfits.

2. **γ**: Gamma (used only for RBF kernel)
   Behaviour: As the value of 'γ' increases the model gets overfits.
   As the value of 'γ' decreases the model underfits.

References:

1. https://towardsdatascience.com/support-vector-machines-svm-c9ef22815589
2. https://www.svm-tutorial.com/2014/11/svm-understanding-math-part-1/
3. https://medium.com/@pushkarmandot/what-is-the-significance-of-c-value-in-support-vector-machine-28224e852c5a
4. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4260380/
5. https://www.intechopen.com/books/advances-in-character-recognition/svm-classifiers-concepts-and-applications-to-character-recognition
6. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72