"Support Vector Machines"

Support Vector machine, abbreviated as SVM. Can be used for both Regression & Classification tasks. But, it is Widely Used in Classification.

→ The Goal of SVM:-

The goal of the SVM algorithm is to Create the best line or decision boundary that Can Segregate n-dimensional space into Classes so that We Can Easily put the new data point in the Correct Category in the future. This Best decision boundary is Called a 'hyperplane'

SVM Chooses the Extreme Point/Vectors that help in Creating the hyperplane. These Extreme Cases are Called as Support Vectors, & hence algorithm is termed as "Supper Vector Machine"

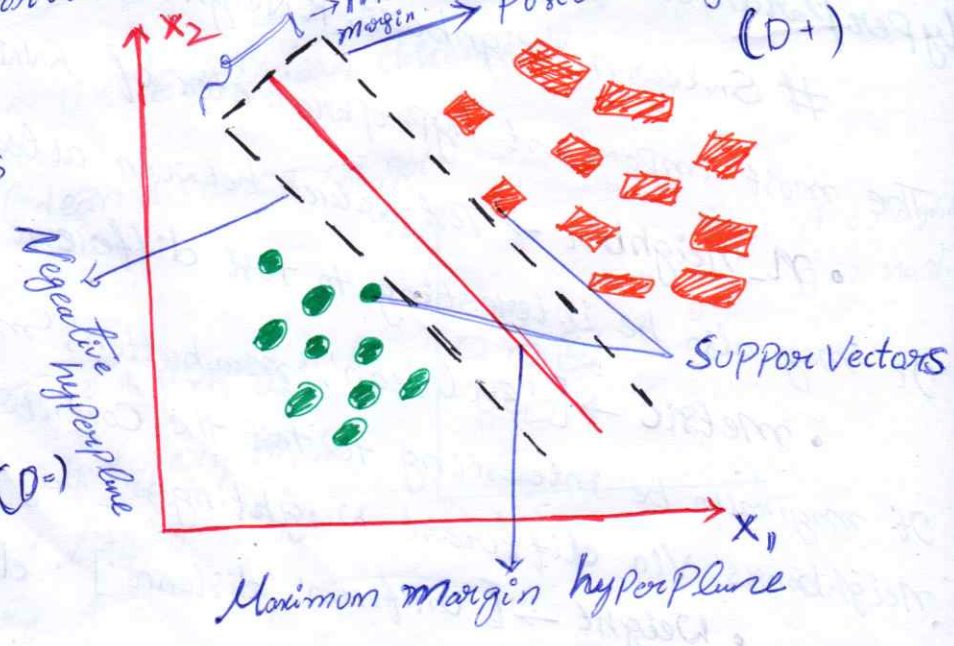
→ Important Concepts in SVM:-

The following are important Concepts in SVM are:-

↳ Support Vectors

↳ Hyper Plane.

↳ Margin.



↳ Hyperplane :-

There can be many possible hyperplanes that could to be chosen, to seperate the two classes of data points. Our Objective is to find a plane that has "maximum margin", i.e, the maximum distance between data points of both classes.

Maximizing the margin distance provides some Reinforcement. So that future data points can be classified with more Confidence.

SVM Selectes the hyper-plane which classifies the classes accurately prior to maximizing margin.

→ First, SVM will generate hyperplanes iteratively that Segregates the classes in best way.

→ Then it will choose the hyperplane that seperates the classes Correctly.

→ The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features, then hyperplane will be a straight line, and if there are 3 features, then hyperplane will be 2-Dimensional plane. It becomes different to imagine when the number of features Exceeds 3.

→ This hyperplane is also called Decision boundary, mathematically We Can Write the Equation of that decision boundary as a line

$$\hat{y}_n = g(x) = w^T x + b = 0 \qquad \hat{y}_n \in \{1, -1\}$$

↳ Support Vectors :-

The data Points or Vectors that are the Closet to the hyperplane & which affect the Position of the hyperplane are termed as Support Vector. Since these Vectors Support the hyperplane, hence Called a Support Vector.

Using these Support Vectors, we maximize the margin of the classifier. If we do some geometry, we can figure out that the distance from any Point to the decision boundary is the following.

$$r = \frac{g(x)}{\|w\|} = \frac{1}{\|w\|}$$

Our goal is to maximize r for the Points closed to the Optimal decision boundary.

↳ Margin :-

It may be defined as the gap between two lines on the closed data Points of different classes.

It Can be Calculated as the Perpendicular distance from the line to the Support Vectors, Given by :-

$$\text{margin} = \frac{2}{\|w\|}$$

Large margin is Considered as a "good margin" & a Small margin is Considered as a "bad margin"

There are two types of margins that are used in SVM.

Hard & Soft:

When the training dataset is linearly seperable, SVM can simply select two parallel lines that maximize the marginal distance; this is called a "Hard Margin"

When the training dataset is not fully linearly seperate, then the SVM allows some margin violation. It allows some data points to stay on the wrong side of the hyperplane or between the margin & hyperplane so that the accuracy is not compromised, this is called a 'Soft margin".

Soft margin permits few of the above datapoints to get misclassified. Also, it tries to make the balance back & forth between finding a hyperplane that attempts to make less misclassifications & maximize the margin

Types of SVM:-

→ Linear SVM: Linear SVM is used for linearly seperable 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, & Classifier is used as Linear SVM.

→ Non-linear SVM:- Non-linear SVM is used for non-linearly seperated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data & Classifier used is called Non-linear SVM Classifier.

Non-Linear Svm:-

→ If data is linearly arranged, then we can seperate it by using a Straight line, but for non-linear data, we cannot draw a straight line.

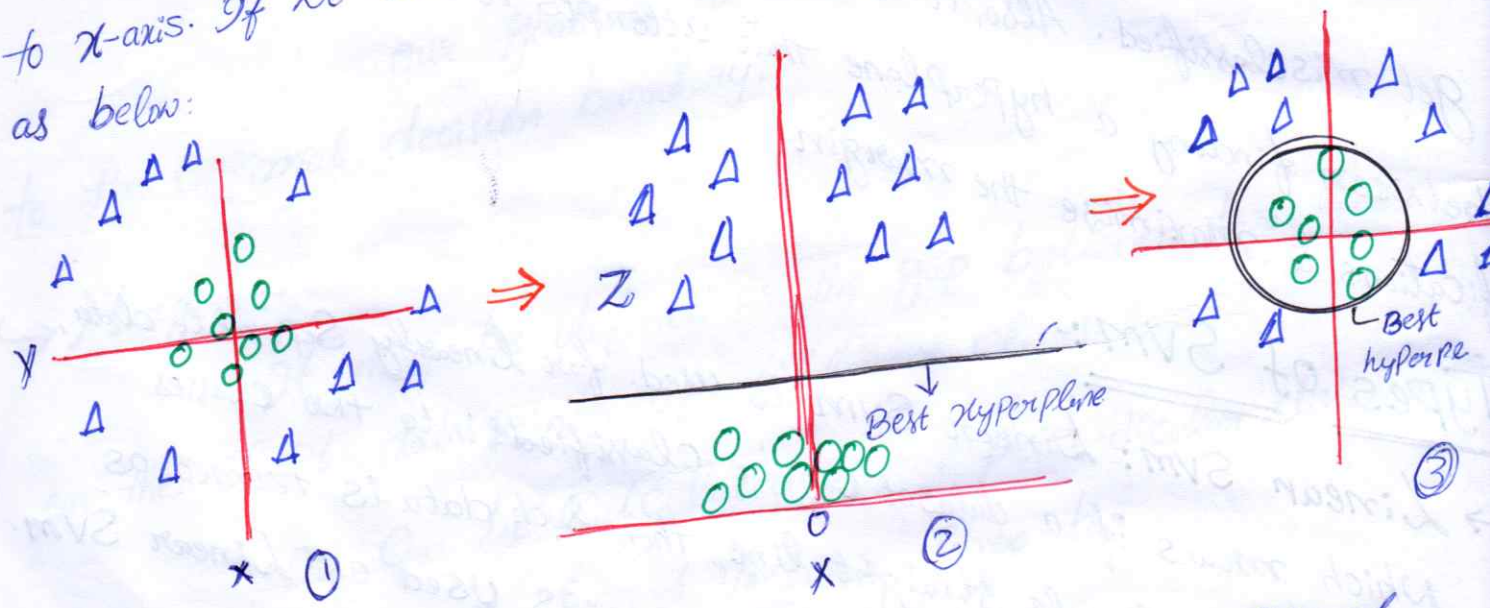
→ So, to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x & y. So for non-linear data, we will add a third demension. It can be calculated as:

$$z = x^2 + y^2$$

→ By adding the third dimension, the sample space will become as image

→ So, now svm will divide the datasets into classes.

→ Since, we are in 3-d space, hence it is looking like a plane parallel to x-axis. If we convert it in 2d space with z=1, then it will become as below:



→ Hence we get a circumference of radius '1' in case of non-linear data.

* All values for z, would be positive, always because z is the squared sum of both x & y.

Nitheesh

SVM KERNELS - KERNEL TRICK :-

In Practice, SVM algorithm is implemented with Kernel that transforms an input data space into the required form. SVM uses a technique called the "Kernel trick" in which takes a low dimensional input space & transforms it into a higher dimensional space.

In Simple words, Kernel Converts non-Seperable Problems into Seperable Problems by adding more dimensions to it. It is mostly useful in non-linear Seperable Problem.

Simply put, it does Some Extremly Complex data transformations, then find out the process to seperate the data based on the labels or inputs you have defined. It makes SVM more Powerful, flexible & accurate. The following are some of the types of Kernals used by SVM:

Kernal = { "linear", "Poly", "RBF", "Sigmoid" }

Linear Kernal :-

It Can be used as a dot Product between any two observations. The formula of linear Kernel is as below:

$$K(x, x_i) = Sum(x * x_i)$$

Polynomial Kernal :-

It is more generalized form of linear Kernal & distinguish Curved or non-linear input Space. Following is the formula for Polynomial:

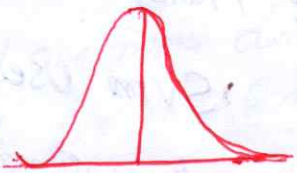
$$K(x, x_i) = 1 + Sum(x * x_i)^d$$

d -> degree of Polynomial -> Specify manually.

Radial Basis Function (RBF) Kernel :-

RBF Kernel, mostly used in Svm Classification, maps input Space in indefinite dimensional Space. Following formula explains it mathematically-

$$K(x, x_i) = Exp(-gamma * Sum(x - x_i^2))$$



→ Here, 'gamma' ranges from 0 to 1. We need to manually specify it in the learning Algorithm.

→ A good default value of gamma is 0.1.

Svm Robust To Outliers :-

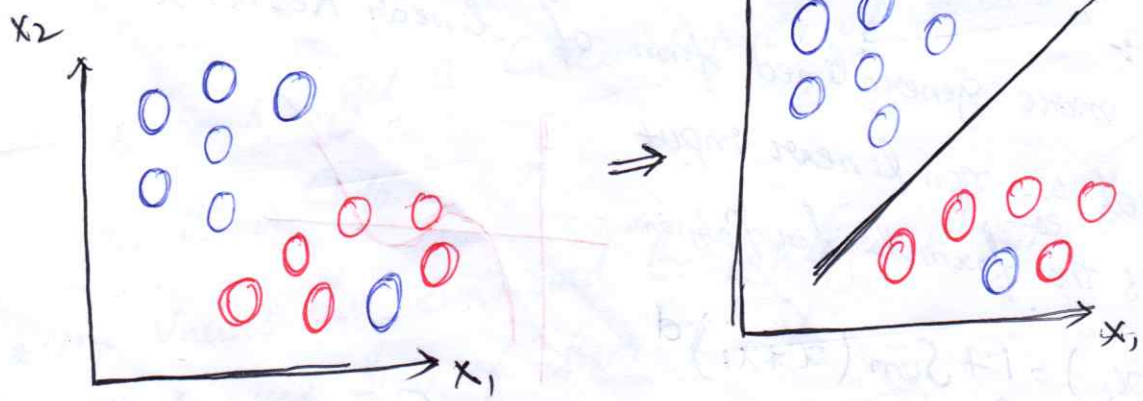
↳ Let Consider a Scenario like shown in below image :-

↳ Here we have one ball in the boundary of the red ball.

↳ So, how does Svm classify the data ? , it's simple : the blue ball in the boundary of red ones is an outlier of blue balls

↳ The Svm algorithm has the characteristic to ignore the outlier & finds the best hyperplane that maximizes the margin.

" Svm is robost to outliers "



→ So, in this type of Data Points, what Svm does, is it finds maximum margin as done with Previous data sets along with that it adds a Penalty each time a Point crosses the margin. Margins in these type of Cases are called Soft margin.

→ When there is a Soft margin to the dataset, the Svm tries to minimize: $(1/\text{margin} + \lambda(\Sigma \text{Penalty}))$.

→ Hinge loss is a Commonly used Penalty. If no violations no hinge loss. If violations hinge loss Proportional to the distance of Violation.

How Svm Works :-

In logistic Regression, we take the output of the linear function & Squash the value within the range [0, 1] using the Sigmoid function. If the squashed value is greater than a threshold value (0.5) we assign it a label 1, else we assign it a label 0.

In Svm, we take the output of the linear function & if that output is greater than 1, we identify it with one class & if the output is -1, we identify it with another class.

Since if $g(x) = 0$, it mean that you testing Sample belongs to the seperating hyperplane & thus there cannot be Some decision about its truth label.

Note that, if $-1 < g(x) < +1$, the the testing datum lies inside the So-Called margin of the classifier.

* Since the threshold Values are Changed to 1 & -1 in SVM, We obtain this Reinforcement Range of Value ([-1, 1]) which acts as margin.

Maths Behind SVM :-

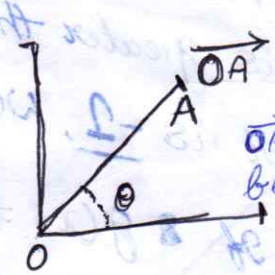
Basic Linear Algerba :-

Support Vector machine basically helps in Sorting the data into two or more Categories with the help of a boundary to differentiate Similar Categories.

So, first let's Revise the formulae for how Each data is Represented in space, & what is an Equation of a line which will help in Segregating the Similar Categories, & lastly the distance formula between a data Point & the line (hyperplane)

Vectors :- (a Point in the Space) :

Vectors are mathematical quantity which has both magnitude & direction. A Point in the 2D plane Can be represented as a Vector between origin & the Point.



$\overrightarrow{OA}$ is vector & length between OA is magnitud

Length of Vectors :-

Length of Vectors are also, Called as Norms. It tells how far Vectors are from the origin.

Length of Vector $x(x_1, x_2, x_3)$ is Calculated as :

$$\|x\| = \sqrt{x_1^2 + x_2^2 + x_3^2}$$

Director of Vectors :-

Direction of Vector $x(x_1, x_2, x_3)$ is calculated as:

$$\left\{ \frac{x_1}{\|x\|}, \frac{x_2}{\|x\|}, \frac{x_3}{\|x\|} \right\}$$

* It is worth noting that the norm of a direction Vector is always Equal to 1. Because of this, the direction Vector w is Called the Unit Vector.

Dot Product :

The dot Product of two Vectors returns a Scalar. It gives us Some insights into how the two Vectors are related.

The geometric formula of a dot Product is defined as:

$$x \cdot y = \|x\| \|y\| \cos(\theta) = \frac{x_1 y_1 + x_2 y_2}{\|x\| \|y\|} \times \|x\| \|y\|$$

$\hookrightarrow \cos(\theta)$ simplified

$$= x_1 y_1 + x_2 y_2$$

This is the algebraic formula of dot Product. In General, dot Product Can be Computed as the following for

$$x \cdot y = \sum_{i=1}^{n} x_i y_i$$

HyperPlane :-

The two dimensional linearly Separable data Can be Seperated by a line. The function of line is

$$y = ax + b$$

We rename x with x_1 & y with x_2 & we get

$$ax_1 - x_2 + b = 0$$

If we define $x = (x_1, x_2)$ & $w = (a, -1)$, we get

$$w \cdot x + b = 0$$

$$\Rightarrow h(x_i) = w^T x_i + b = 0$$

This is Equations of hyperplane.

$w^T \rightarrow$ Unit vector.

$x_i \rightarrow i^{th}$ training Example.

$y_i \rightarrow i^{th}$ label corresponding to x_i

Classifier :-

Once, we have the hyperplane, we can then use the hyperplane to make Prediction, we define the hypothesis function h as :-

$$h(x_i) = \begin{cases} +1 & w \cdot x_i + b \geq 1 \\ -1 & w x_i + b \leq -1 \end{cases}$$

Combining above two Equation, it can be written as.

$$\hat{y}_i (w \cdot x_i + b) - 1 \geq 0 \quad y_i \in (-1, 1)$$

The Point above hyperplane will be classified as Class +1 (Positive class) & the Point below the hyperplane will be classified as Class -1 (negeative class)

$$\pi_{-ve} = w^T x_i + b = -1 \rightarrow (1)$$

$$\pi = w^T x_i + b = 0$$

$$\pi_{+ve} = w^T x_i + b = +1 \rightarrow (2)$$

But here the signs were about training data, which is how we are training our model. That for positive class, give a positive sign & for negeative give a negative sign.

But while testing this model on test data, If we predict a positive class (positive sign) correctly as positive, then two positive makes positive & hence greater than zero result. The same applies if we correctly predict the negative group since two negatives will again make a positive.

But, if the model miss classifies the positive group as a negative group then one plus & one minus makes a minus, hence overall less than zero.

Summing up the abovt concept :-

"The Product of a predicted & actual label would be greater than zero on correct prediction, otherwise less than zero"

* W^T → Transpose of (W).

$$\hat{y}_i [W^T . x_i + b] = \begin{cases} \geq 0 & \text{if correct} \\ < 0 & \text{if incorrect} \end{cases}$$

Margin :-

Margin is nothing but distance between two classes, therefore subtracting Eq ② & ①, that gives:

$$W^T x_{i_{+ve}} - W^T . x_{i_{-ve}} = 2$$

$$\Rightarrow W^T (x_{+ve} - x_{-ve}) = 2.$$

$$\Rightarrow \frac{w^T}{\|w\|}(x_{+ve} - x_{-ve}) = \frac{2}{\|w\|}$$

$$\Rightarrow \text{margin} = \boxed{2/\|w\| = M}$$

Here $\|w\|$ is the Euclidean norm for the length of w given by:

$$\|w\| = \sqrt{w_1^2 + w_2^2 + w_3^2 \ldots + w_n^2}$$

Taking distance to any one of the Plane/side from the hyperplane is given by:-

$$\boxed{d = 1/\|w\|}$$

Optimization:- (Primal & Dual) :-

The Goal of Svm is to get a margin with maximum distance. Therefore, in order to find the optimal margin, What we want to do is to maximize d. With these Conside Considerations, we can finally state the optimization Problem for the model.

$$d = \max\left(1/\|w\|\right) \Rightarrow \min \|w\|.$$

Subject to $\min_n |w^T x_n + b| = 1$

Well, from Calculus we know that in order to find a Maximum Value, We need to use derivates, to make things for us When computing derivatis, ~~We~~ ~~Write~~ it as:-

and, as l_2 optimization are often more stable than l_1 optimization so again above Equation can be written as :-

$$\min \frac{\|w\|^2}{2} \quad \text{such that} \quad y_i(w^T x_i + b) - 1 \geq 0$$

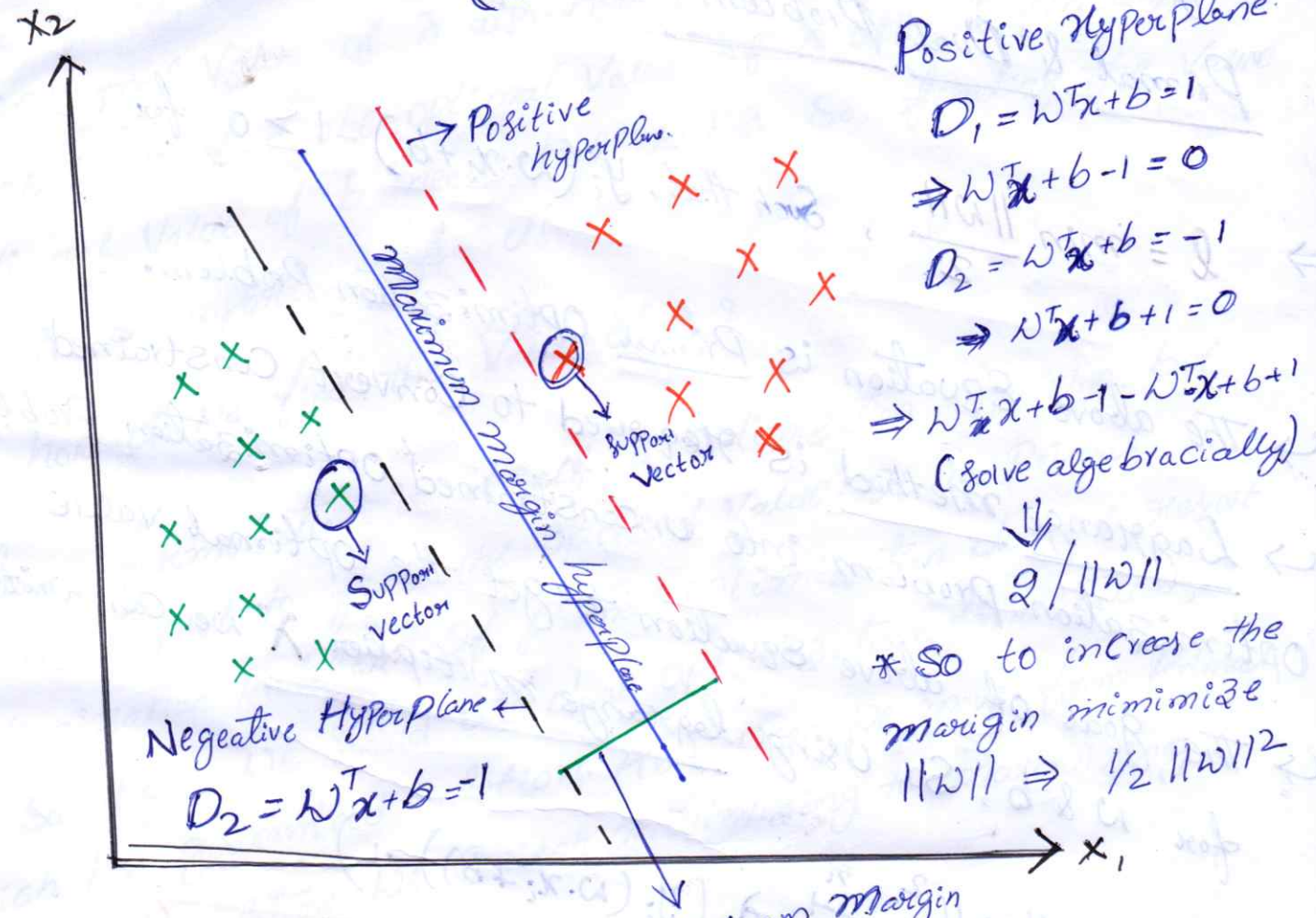
(separating hyperplane)

↓ Maximize margin

↓ This is called as Hard margin SVM (Support Vector formulation)

(Convex Quadratic optimization)

To Sum-UP :-



Positive Hyperplane

$D_1 = w^T x + b = 1$

$\Rightarrow w^T x + b - 1 = 0$

$D_2 = w^T x + b = -1$

$\Rightarrow w^T x + b + 1 = 0$

$\Rightarrow w^T x + b - 1 - w^T x + b + 1$

(Solve algebraically)

$2/\|w\|$

* So to increase the margin minimize $\|w\| \Rightarrow 1/2 \|w\|^2$

In layman term Svm optimization :-

if $y_i(w \cdot x_i + b) - 1 = 0$

then x_i, y_i are Support Vectors

Elif, $y_i(w \cdot x_i + b) - 1 > 0$:

then Save the Parameters w, b

Elif, $y_i(w \cdot x_i + b) - 1 < 0$:

then Update the Parameters w, b

⇒ We have to find the Values of Parameters (w, b), which minimizes $\|w\|$ for optimal hyperplane.

⇒ The Problem of finding the Values of w & b is called an "optimization Problem."

⇒ Any optimization Problem can be formulated in two way "Primal & Dual Problem"

⇒ $l = \min \frac{\|w\|}{2}$, Such that, $y_i (w \cdot x_i + b) - 1 \geq 0$ for $i = 1 \ldots l$.

↪ The above Equation is Primal optimization Problem.

↪ Lagrange method is Requried to Convert Constrained optimization Problem into Unconstrained optimization Problem.

↪ The goal of above Equation to get the optimal value for w & b, So Using "lagrange multiplier λ" we can write.

$$l_p = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \lambda_i \left(y_i (w \cdot x_i + b) - 1 \right)$$

$$= \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \lambda_i \left(y_i (w \cdot x_i + b) \right) - \sum_{i=1}^{n} \lambda_i$$

⇒ $\partial l / \partial w = \nabla_w l (w, b, \lambda) = w - \sum_{i=1}^{n} \lambda_i y_i \cdot x_i = 0 \rightarrow (1)$

from eq (1) $\Rightarrow w = \sum_{i=1}^{n} \lambda_i y_i x_i \rightarrow (*)$

$$\Rightarrow \partial l/\partial \lambda = \nabla_\lambda l(w,b,\lambda) = \sum_{i=1}^{n} y_i (w \cdot x_i + b) - 1 = 0 \rightarrow (2)$$

$$\Rightarrow \partial l/\partial b = \nabla_b l(w,b,\lambda) = -\sum_{i=1}^{n} \lambda_i \cdot y_i = 0 \rightarrow (3)$$

~~$\Rightarrow \partial l/\partial w = \nabla_w l(w,b,\lambda) = w - \sum_{i=1}^{n} \lambda_i y_i x_i = 0 \rightarrow (1)$~~

As, from the above formulation, we only able to find the optimal Value of w, & that is to dependent on λ, So we need to find the optimal Value of λ also. And finding optimal Value of b needs to find the optimal Value of λ also, & finding optimal Value of b needs both w & λ. So, finding the Value of λ is important for us.

So how do we find the Value of λ?

Above, formulation is itself a optimization algorithm, but it not helpful to find the optimal Value. It is Primal optimization Problem. If Primal optimization doesn't result in Solution, we should use dual optimization formulation, Which has guaranteed Solution. Also when we move from Primal to dual formulation we switch minimizing to maximizing the loss function.

We will divide the Complete formulation into three Parts to Easier to Understand:

↳ Promblem formulation & Substitution Value from Primal

↳ Simplify the loss function Equation after Substitution

↳ final optimization formulation to get the Value of λ.

$$l_{dual} = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \lambda_i \left(y_i (w \cdot x_i + b)\right) + \sum_{i=1}^{n} \lambda_i$$

Substituting the Value of $w = \sum_{i=1}^{n} \lambda_i y_i x_i$ into above Equation,

$$l_d = \frac{\sum_{i=1}^{n} \|\lambda_i \cdot y_i \cdot x_i\|^2}{2} - \sum_{j=1}^{n} \cdot \sum_{i=1}^{n} \lambda_j y_j \left(\|\lambda_i \cdot y_i x_i\|\right) \cdot x_j + b) + \sum_{i=}^{n} \lambda$$

* Above Equation is Called Dual optimization Problem.

$$\Rightarrow l_d = \frac{\sum_{i=1}^{n} (\lambda_i \cdot y_i \cdot x_i)^T \cdot (\lambda_i y_i \cdot x_i)}{2} - \sum_{j=1}^{n} \cdot \sum_{i=1}^{n} \lambda_j \lambda_i \cdot y_j \cdot y_i + x_i x_j + \sum_{j=1}^{n} \cdot \sum_{i=1}^{n} \lambda_j y_j - \sum_{i=1}^{n} \lambda_i$$

Since Constriant is $\lambda_j \geq 0$, $\sum_{i=1}^{n} \lambda_j \cdot y_i = 0 \Rightarrow$ 3rd form become Zero

$$\Rightarrow l_d = \sum_{i=1}^{n} \lambda_i - \frac{\sum_{i,j=1}^{n} \lambda_j \cdot \lambda_i \cdot y_j \cdot y_i \cdot x_i \cdot x_j}{2}$$

In simplified Equation of above dual optimization Promblem

$$\max l_d = \sum_{i=1}^{n} \lambda_i - \frac{\lambda^T \cdot \lambda K}{2}$$

Where, $K(i,j) = y_i y_j \cdot x_i \cdot x_j \rightarrow$ In vector, $K = y^T y \cdot x^T \cdot x$

↓
nothing but dot Product of Input Variable x

Once we get λ from above, we Can get w & b.

* Dual forms - Rewrites the Same Problem using a different Set of Variables, So the alternate formulation will help in Eliminating the dependence & reducing the Effect will be done by Kernelization

* Lagrange Multiplier Method :- It is a Strategy to find the local minima or maxima of a function subject to the Condition that one or more Equations have to be Statisfied Exactly by the Chosen Values of Variables.

Till now we have discussed, linear Classification - Hard margin, no data Points are allowed in the margin areas.

The Problem with Hard margin SVM is that it only Works for linearly Seperable data also leads to Underfitting.

* We can simply use optimization libraries CVXOPT to solve optimal Parameters

Soft Margin SVM - Optimization :-

The Problem with Hard margin SVM is that it only Works for linearly Seperable data, However, this would not be the Case in the real World. It is most likely that in Practical Cases the data will Contain Some noise & might not be linearly Seperable. We'll look at how Soft margin SVM handles this Problem

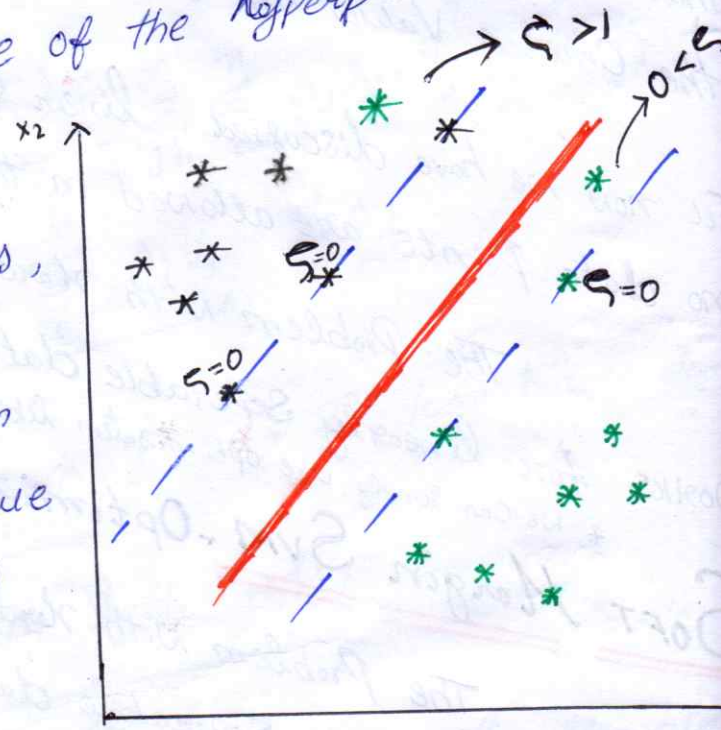
Basically, the trick Soft margin SVM is using Very Simple, it adds Slack Variables ξ_i to the Constraints of the optimization Problem. The Constraints now become :

$y_i(w \cdot x_i + b) \geq 1 \longrightarrow$ Hard margin optimization

Now, the Constraints become :-

$$y_i (w \cdot x_i + b) \geq 1 - \zeta_i \quad ; \quad i = 1 \ldots n$$

* Here, the Slack Variable is defined as a tolerance Variable. Points that lie on the margin will have $\zeta = 0$. on the other hand, the farther we are from the correct boundar the larger the value of ζ will be. Ultimately, for Points that are on the wrong side of the hyperplane, we Expect ζ to be greater than 1.



* By adding the Slack Variables, When minimizing the objective function, it is possible is we can always Choose a large Enough value of ζ. So that all the Examples will Stasify the Constraints.

* One technique to handle this is to use Regularization. For Example, we Could use L_2 Regularization to Penalize large value of ζ. The Regularized optimization Problem becomes:

$$\min_{w,b,\zeta} \frac{1}{2} \|w\|^2 + \sum_{i=1}^{n} \zeta_i$$

$$\text{Subject to } y_i (w \cdot x_i + b) \geq 1 - \zeta_i \quad i = 1 \ldots n$$

Also, We Want to make Sure that We do not minimize the objective function by Choosing negative Values of ξ. We add the Constraints "$\xi_i \geq 0$".

We also add a "regularization Parameter C" to determine how important ξ should be, which means how much We want to avoid misclassifying Each training Example. The regularized optimization Problem becomes.

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

Subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \; ; \; \xi_i \geq 0, \; i = 1 \ldots n$

By Considering these new Variables in the original Primal Problem We get:

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\lambda_i[y_i(wx_i + b) - 1 + \xi_i] - \sum_{i=1}^{n}\mu_i\xi_i$$

Here, μ is a lagrange multiplier We have defined for the tolerance Values of ξ. We Except Each μ greater than or Equal to Zero.

By doing all the hard math, the optimization Problem Could be transformed to a dual Problem.

$$\max_{\lambda} \sum_{i=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{i=1}^{n}\lambda_i\lambda_j y_i y_j x_i \cdot x_j$$

Subject to $0 \leq \lambda_i \leq C, \; i = 1 \ldots n, \; \sum_{i=}^{n}\lambda_i y_i = 0$

Here the Constraint, $\lambda_i \geq 0$, has been Changed to $0 \leq \lambda_i \leq C$.

Regularization Parameter C:

C is the Parameter which Controls the trade-off between the length of margin & number of the misclassificat on the training data, in other words, it gives you control of how the SVM will handle Errors.

For C=0, it is not letting any misclassification to occur which is nothing but hard margin classification & it result in narrow margin.

For C>0, it mean no more than C observation can violate the margin, as C increases margins also widens.

The Correct Value of C is ~~decided~~ decided by Cross-Validation & it is this Parameter only, that can result in "bias-Variance trade-off in SVM"

If the Value of C=0, which results in high Variance but if the Value of C=∞ it results in high-bias.

Loss-Function :-

Let us Consider, the Value of ξ_i for the case of C≠0 Because the objective will always try to minimize ξ_i as much as Possible, the Equation must hold as an Equality & We have,

$$\zeta_i = \begin{cases} 1 - y_i(w \cdot x_i + b) & \text{if } y_i(w \cdot x_i + b) < 1 \\ 0 & \text{if } y_i(w \cdot x_i + b) \geq 1 \end{cases}$$

This is equivalent to the following closed form:

$$\zeta_i = \max(1 - y_i(w x_i + b), 0)$$

If we plug this closed form into the objective of our SVM optimization problem, we obtain the following unconstrained version as loss function & regularizer.

$$\min_{w,b} \underbrace{\frac{\|w\|^2}{2}}_{l_2 \text{ regularizer}} + C \sum_{i=1}^{n} \underbrace{\max[1 - y_i(w \cdot x_i + b), 0]}_{\text{Hinge-loss}}$$

This formulation allows us to optimize the SVM parameters (w, b) just like logistic regression (eg. through gradient descent). The only difference is that we have hinge loss instead of the logistic loss.

Extensions of SVM:-

* The model produced by support vector classification depends only on a subset of traing ~~data~~, because the cost function for building the model does not care about training points that lie beyond the margin.

Before moving on, it's worth pointing out that SVM's are among the most powerful machine learning algorithms for classification tasks and are used extensively for applications ranging from "Computer Vision to NLP".

* Support-Vector Clustering → UnSupervised learning

* Multi-Class Svm → One vs one or One vs rest

* Transdective Svm → Semi-Supervised learning.

* Regression → Support Vector Regressor

* Bayesian Svm → Data Augmentation.

Pros of Svm:-

↳ High Stability due to dependency on Support Vectors & not the data Points

↳ Does not get influenced by outliers

↳ No assumptions made of the datasets

↳ Numeric Predictions Problem Can be dealt with Svm

Cons of Svm:-

↳ BlackBox method

↳ Inclined to Overfitting method

↳ Very Rigorous Computation

↳ Svm doesn't directly Provide Probability Estimates, these are Calculated Using an Expensive five-fold Cross-Validation.

HyperParameters of SVC :-

Sklearn.Svm.SVC (C=1.0, Kernal='rbf', degree=3, gamma='scale')

- C=1.0 → Regularization Parameter → Squared L_2 Penality.
- Kernal='rbf' → Kernal type
- degree=3 → degree of Polynomial Kernal function
- gamma='scale' → Kernel Coefficient

⇒ Kernals in ['linear', 'Poly', 'rbf', 'Sigmoid']
↳ default

⇒ C in [100, 10, 1.0, 0.1, 0.001] :

The C regularization Parameters accept Values from you to allow a certain level of misclassification in Each training data set. Higher C regularization Values lead to small margin hyperplane & do not allow much misclassification. Lower Values, on the other hand, leads to high-margin & greater misclassification.

→ Gamma in [1, 0.1, 0.01, 0.001, 0.0001]

Gamma is a hyperParameter used with non-linear Svm. One of the most Commonly used non-linear Kernals is the Radial basis function (RBF). Gamma Parameter of RBF Controls "the distance of the influence" of a single training Point

Low Value of gamma indicate a large Similarity Radius which results in more Points being grouped together. For high Values of gamma, the Points needs to be Very close to Each other in order to be Considered in the group (or class)

Therefore, models with very large gamma value tend to overfit.

In other words, The gamma Parameter defines the range of support vectors that will impact the positioning of the hyperplane. High gamma value considers only nearby data points & low value considers far away points.

gamma : {'scale', 'auto'} or float

'scale' → default → 1/n_features * X.var()

'auto' → 1/n_features

Gamma Vs C Parameter:

For a linear kernel, we just need to optimize the 'C' Parameter. However, if we want to use an RBF kernel, both C & gamma Parameters need to optimized simultaneously. If gamma is large, the Effect of C becomes negligible.

If gamma is small, C affects the model just like how it affects a linear model. Typical values for C & gamma are as follows:

$0.0001 < gamma < 10 \quad ; \quad 0.1 < C < 100$

Hyper-Tuning :-

```
Param_grid : {'C' : [0.1, 1, 10, 100, 1000],
              'gamma' : [1, 0.1, 0.01, 0.001, 0.0001],
              'Kernel' : ['rbf']
             }
```