S21CSE471: Statistical Methods in AI 1^{st} **Feb 2021**

Support Vector Machine

Prepared by: Team 7 - Abhijeet Ashish Darshit

1 Introduction

Support Vector Machine is a supervised learning algorithm with an objective to find a hyperplane in an N-dimensional space that distinctly analyzes the data for classification. SVM is one of the most robust prediction methods used for prediction. Originally it was Developed at AT&T Bell Laboratories by Vapnik and his co-workers and successively extended by several other researchers.

An SVM maps binary labeled training data to points in space to maximize the width gap between the two categories. Future data are then mapped into the same space and predicted to belong to a which category based on which side of the gap they fall.

What is support vector machine?
Support Vector Machine is a supervised learning algorithm with an objective to find a hyper-plane in an N-dimensional space that distinctly analyzes the data for classification.Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin).

SVM has its solid mathematical foundation in statistical learning theory, SVMs have demonstrated highly competitive performance in numerous real-world applications, such as bioinformatics, text mining, face recognition, and image processing, which has established SVMs as one of the state-of-the-art tools for machine learning.

Let's take an example of linear classification in 2D to understand it in a simpler way.

Suppose we have a 2 labeled classes red circle and blue circle on a graph which is separated by a hyper-plane (Fig 1). This hyper-plane is basically a line here which dividing a plane in two parts where each class is lay in either side.

Any point that is in a left side of the plane is belongs to red circle class and on right it belongs to blue circle class. Make it in a mathematical way, our equation for Hyperplane is - $w^T x = 0$

So, for our 2D line it will be, $w_0 + w_1 * x_1 + w_2 * x_2 = 0$
Let's put the value $w_0 = -6, w_1 = 1, w_2 = 1,$
Our equation of line will be,$x_1 + x_2 - 6 = 0 \implies x + y – 6 = 0$

If we put the coordinates of any point in above equation and if the value is +ve it falls under blue class and if the value is –ve falls into red class. And if the value is 0 it's on the line. So, the sign is the classification rule here for the binary class.

$$f(x) = \text{sign}\left(w^T x\right)$$

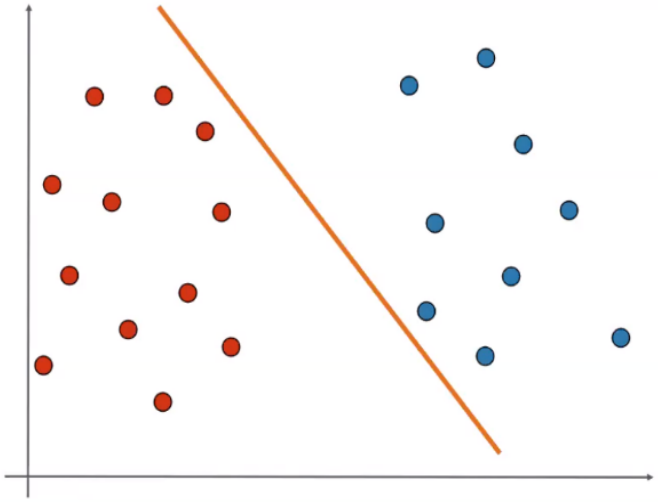

Figure 1: Data Separated by hyperplane

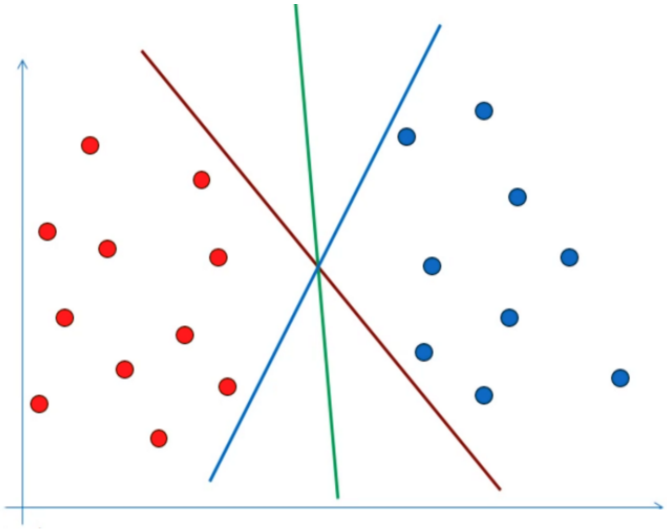

Figure 2: Which Line is better

2 Selection of Hyperplane

Now, the questions arise for the selection of hyperplane. How we select the best hyperplane?

In the above figure the green line looks like the best solution as it would give better prediction on future dataset. Because of the width of band around the boundary called margin.

To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find plane that has the maximum margin (Fig - 2), 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.

As we can see in fig-3 some points are supporting the boundary of margin line. These points or samples

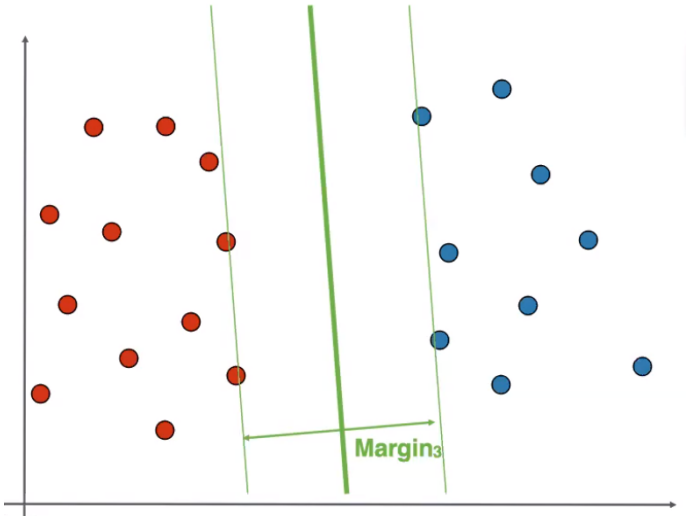

Figure 3: Hyperplane maximizes margin

which supports the boundary of margin lines are called **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. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

3 Vapnik–Chervonenkis dimension

Vapnik and Chervonenkis has provide some theory about why it is necessary to maximize the margin. As they suggested that we can bound the future test performance (i.e future testing error R(α)) on the basis of training error (Rtrain (α)) and VC-dimension (monotonous function).

$$R(\alpha) <= R_{\text{tein}}\ (\alpha) + \frac{\sqrt{f \mid h}}{N}$$

Where h is the VC dimension, and f(h) given by:

$$f(h) = h + h\log(2N) - h\log(h) - c$$

To reduce test error, keep training error low (say 0), and minimize the VC-dimension, h.
Assuming that all our points contained within the circle of diameter D. So, our relative margin will be:

$$\text{Relative Margin}: \frac{\rho}{D}$$
$$\text{VC} - D, h \leq \min\left(h, \left\lceil \frac{D^2}{\rho^2} \right\rceil\right) + 1.$$

where ρ is the margin

The VC dimension, h was written as a minimum of the inverse of relative margin square and dimension of the data. If we could maximize the relative margin, we would be minimizing its inverse square, and if that goes below dimensions of data, h will become independent of dimension.

So, the observations made on the basis of above equation that it proves us the larger margin leads to a smaller VC-Dimension, which leads to better bound on generalization error. In general, we can say the larger the margin lowers the generalization error of the classifier.

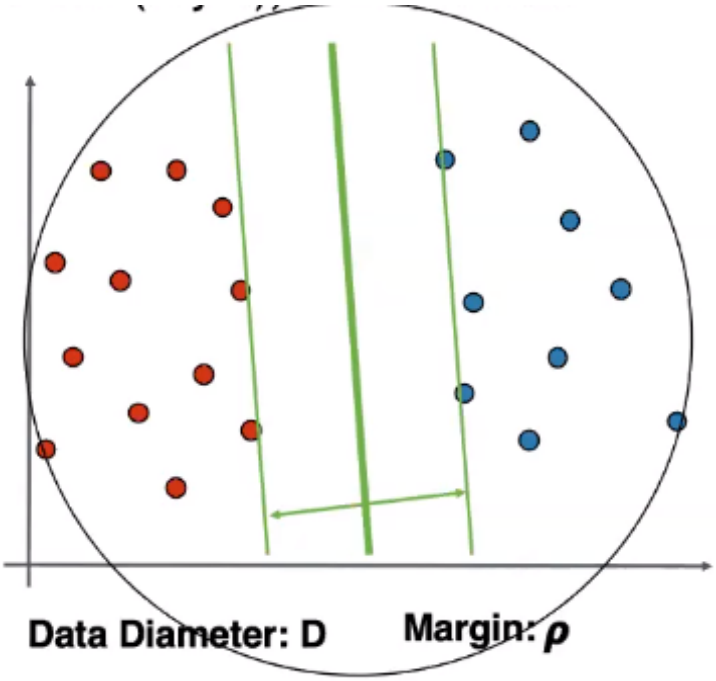

Figure 4:

4 Formulation of Margin

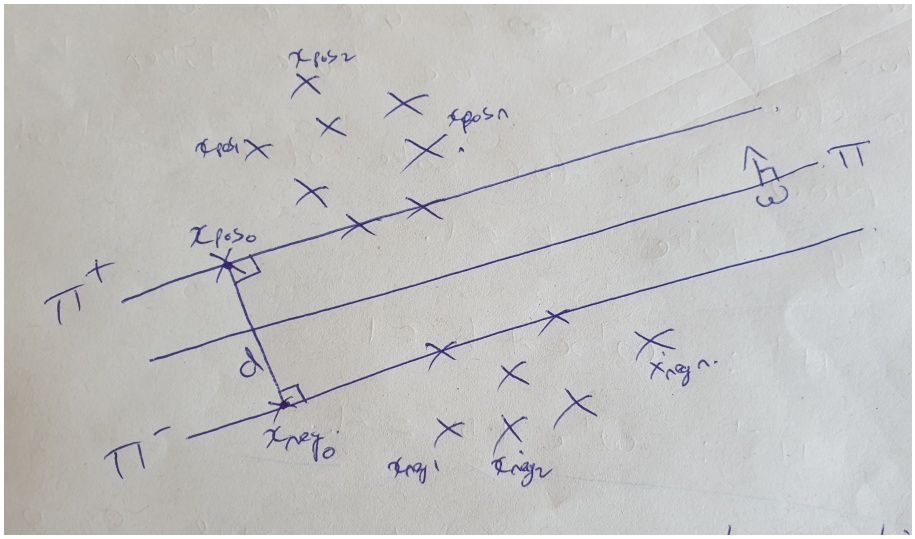

Figure 5:

Here π^+, π^- and π are hyper-planes which are parallel to each other. $\vec{w}$ is perpendicular to all this hyper-planes. we know equation of hyper-plane is $w^T\mathrm{x} + \mathrm{b}$. so let,

$$\pi^+ \colon\ w^T\mathrm{x} + \mathrm{b} = \mathrm{k}$$

$$\pi^-\text{: } w^T\text{x} + \text{b} = \text{-k}$$

$$\pi\text{: } w^T\text{x} + \text{b} = 0$$

Here $\vec{w}$ = normal vector to hyper plane and b = distance of plane from origin.

One question we might get from above hyper-plane equations is, why we have taken k and -k , why can't we take it like k1 and -k2?

We know that π^+ and π^- are equidistant from π so we can take values as k and -k.

Now let us take k=1 for now to make formulation of margin easy mathematically,

$$\pi^+\text{: } \vec{w}\text{x} + \text{b} = 1$$

$$\pi^-\text{: } \vec{w}\text{x} + \text{b} = \text{-1}$$

Now let us derive equation of margin. As shown in figure x_{pos0} and x_{neg0} are 2 points on π^+ and π^- respectively.

$$\pi^+\text{: } wx_{pos0} + \text{b} = 1$$

$$\pi^-\text{: } wx_{neg0} + \text{b} = \text{-1}$$

Now our margin is nothing but shortest distance between π^+ and π^- . Points x_{pos0} and x_{neg0} also lies on perpendicular vector to π^+ and π^- so our margin is distance between x_{pos0} and x_{neg0} .

$$\text{d} = \text{Margin} = w^T(x_{pos0} - x_{neg0}) = 2$$

Divide on both sides with L2 norm of $\vec{w}$.

$$\frac{w^T(x_{pos0} - x_{neg0})}{L2normof\vec{w}} = \frac{2}{L2normof\vec{w}}$$

Now $\frac{w^T}{L2normof\vec{w}}$ is unit vector. So LHS term is nothing but margin.

$$\text{Margin} = \text{d} = \frac{2}{L2normof\vec{w}}$$

Now for better prediction using sum we want this margin to be as much maximum as possible. So now our objective function is,

$$\max(\text{d}) = \max(\frac{2}{L2normof\vec{w}})$$

Lets assume w*,b* are values of hyper-plane for which margin is maximum so we can write objective function as,

$$(\text{w*,b*}) = \underset{(w,b)}{argmax}(\frac{2}{L2normof\vec{w}})$$

Now to complete our formulation at sum only one thing is remaining that are constraints.Let get them,

For all x_{posi}; which are on π^+ or above it ,we can say that

$$w^T x_{posi} + b > 1 \text{ , for every } x_{posi} \text{ which are above } \pi^+$$

$$w^T x_{posi} + \text{b} = 1 \text{ , for every } x_{posi} \text{ on } \pi^+$$

In general for x_{posi}; which are either on π^+ or above it we can say $wx_{posi} + \text{b} \geq 1 - eq5$

Similarly for x_{negi}, that is point either on π^- or below it we can say,

$$w^T x_{negi} + b < 1 \text{ , for every } x_{negi} \text{ which are below } \pi^-$$

$$w^T x_{negi} + \text{b} = 1 \text{ , for every } x_{negi} \text{ on } \pi^-$$

In general for x_{negi}; which are either on π^- or below it we can say $wx_{posi} + \text{b} \leq -1 - eq6$

Now eq 5 and eq 6 are our constraints as you can see both are very similar. So why should we write 2 constraints we can combined both of them for that lets define a variable y_i; $y_i = 1$ for every x_{posi} and y_i = -1 for every x_{negi}.

Multiplying y_i to eq 5 and eq 6, we get following 2 equations

$$y_i * (w^T x_{posi} + b) \geq 1$$

$$y_i * (w^T x_{negi} + b) \geq 1$$

so our final constraint is, $y_i * (w^T x_{negi} + b) \geq 1$ where $y_i = 1$ for every x_{posi} and y_i = -1 for every x_{negi}.

Formulation of SVM is,

$$\text{Objective Function: (w*,b*)} = \underset{(w,b)}{argmax}(\frac{2}{L2normof\vec{w}})$$

$$\text{Constraint: } y_i * (w^T x_{negi} + b) \geq 1 \text{ for every i.}$$

This formulation is also called hard margin SVM as we are assuming our data is linearly separable.

Now one thing to note here is above derived formulation can be applied to SVM only when our data is linearly separable. This is because of the constraint we have that every x_{posi} is either on plane π^+ or above it and is either on plane π^- or below it.

Now lets see what happen when given data is not linearly separable.

As we can see from above diagram that our data-set is not linearly separable but it is almost linearly separable except of few points, most the points are perfectly placed in respective regions. So in such data-set we will never be able to find w*,b* using our hard margin SVM formulation as it will always say there is no w*,b* that satisfies our constraint. So in order to solve such almost linearly separable data-set and find hyper-plane we have to modify our hard margin SVM.

In order to do this let us define a new variable α_i whose job is to differentiate 'correctly classified points' from 'incorrectly classified points'. $\alpha_i = 0$ for every correctly classified points, in other words for every i

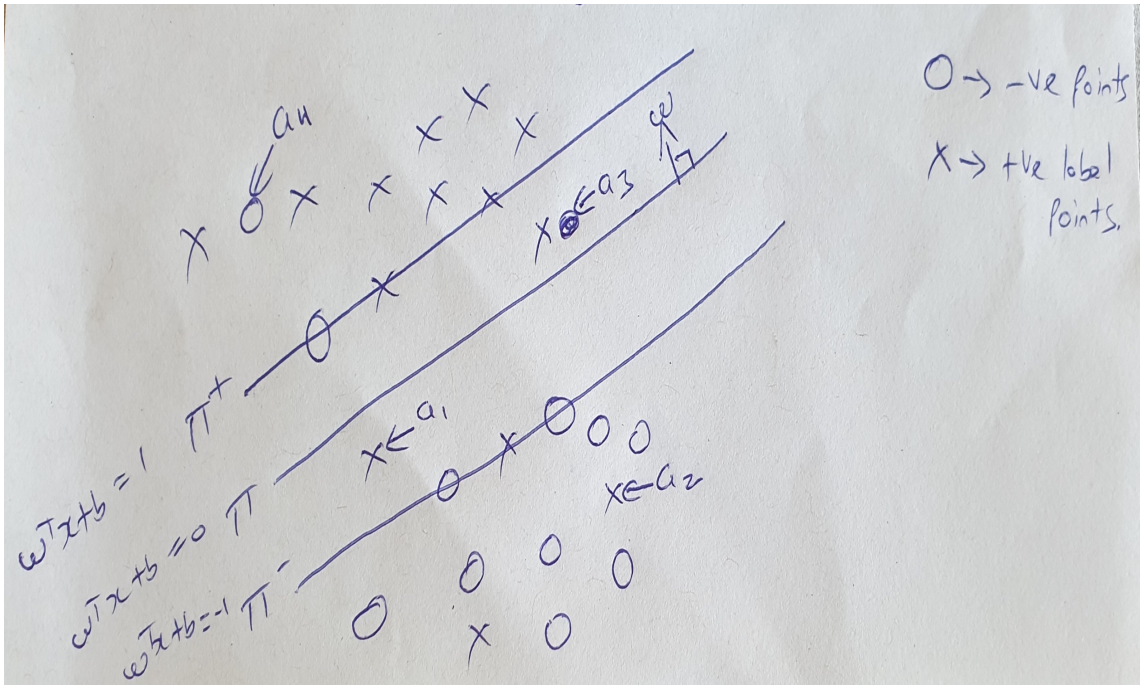

Figure 6:

where $w^T x_i + b \geq 1$ and $\alpha_i > 0$ for every incorrectly classified points, in other words for every i where $w^T x_i + b < 1$

Lets first understand below 2 results:

1. $y_i * (w^T x_i + \text{b}) \geq 1$ for all correctly classified, in derivation of hard margin SVM we saw that for every correctly classified point eq $w^T x_i + b \geq 1$ is satisfied.

2. Now to understand how all incorrectly classified points are covered by inequality $y_i * (w^T x_i + b) \leq 1$ we will have to take 3 points into consideration a1,a2,a3 as show in figure, all this positive label points are outside positive label region.

points	$w^T x_i + b$	y_i	$y_i * (w^T x_i + b)$
a_1	$-1 < w^T x_{a_1} + b < 0$	$y_{a_1} = 1$	$-1 < y_{a_1} * (w^T x_{a_1} + b) < 0$
a_2	$w^T x_{a_2} + b < -1$	$y_{a_2} = 1$	$y_{a_2} * (w^T x_{a_2} + b) < -1$
a_3	$0 < w^T x_{a_3} + b < 1$	$y_{a_3} = 1$	$0 < y_{a_3} * (w^T x_{a_3} + b) < 1$

So from above we can see that any incorrectly place positive points has value of $y_i * (w^T x_i + b) < 1$

Similarly by taking 3 incorrectly place negative points in respective region we can find that all incorrectly place negative points also has value of $y_i * (w^T x_i + b) < 1$.

For all points outside its respective label region value of α_i gives measure of how much this point is away from it's correct plane. For eg: for incorrectly classified point a_1 value of α_1 says how much point a_1 is away from pie+(as a_1 is positive label point).

And for points which are correctly placed in there respective labeled region value of α_i is always 0.

Here note one thing that α_i always has a non-negative value.

Now let us use this α_i to modify our Hard margin SVM. In our linearly separable data-set we knew that all points satisfy $y_i * (w^T x_i + b) \geq 1$, we have even seen how this is true at the time of derivation.

Similarly for almost linearly separable data-set our constraint is, $y_i * (w^T x_i + b) \geq 1 - \alpha_i$, where $\alpha_i = 0$ for correctly classified points and $\alpha_i > 0$ for incorrectly classified points.

Let us take few incorrectly classified data points and check wether they satisfy constraints or not. lets take a1,a2 and a3.

Lets assume a1,a2 and a3 lies on below hyper-plane equation.

a1: $y_a 1 * (w^T x_{a1} + b) - 0.5 = 1 - (1.5)$

a2: $y_a 2 * (w^T x_{a2} + b) - 1.5 = 1 - (2.5)$

a3: $y_a 3 * (w^T x_{a3} + b) 0.5 = 1 - (0.5)$

As we can see we are able to represent any incorrectly classified data point in form of $y_i * (w^T x_i + b) = 1 - \alpha_i$. Hence we can say all the points in data whether it is correctly classified or incorrectly classified they follows constraint $y_i * (w^T x_i + b) \geq 1 - \alpha_i$.

Now lets define our objective function,

$$\text{For Linearly separable dataset : (w*,b*)} = \underset{(w,b)}{argmax}(\frac{2}{L2normof\vec{w}}) = \underset{(w,b)}{argmin}(\frac{L2normof\vec{w}}{2})$$

$$\text{For non-Linearly separable dataset : (w*,b*)} = \underset{(w,b)}{argmin}(\frac{L2normof\vec{w}}{2}) + \frac{c}{n} * min(\sum \alpha_i)$$

So below if our final formulation,

$$\text{Objective function: (w*,b*)} = \underset{(w,b)}{argmin}(\frac{L2normof\vec{w}}{2}) + \frac{c}{n} * min(\sum \alpha_i)$$

$$\text{constraint: } y_i * (w^T x_i + b) \geq 1 - \alpha_i$$

This formulation is also called soft margin SVM as our data is not linearly separable.

Our objective function was to maximize margin in linearly separable data-set but in almost linearly separable data-set our objective function is to maximize margin and also to minimize error which is nothing but to minimize incorrectly classified points. so we are suppose to min$\sum \alpha_i$ that is minimize sum of α_i for all incorrectly classified points.

'c' in above equation is our hyper-parameter. If c is very large then we will over-fit our training data-set as significance of first term in above formulation equation will be negligible. If c is small then we will under-fit our training data-set as significance of second term in above formulation equation will be negligible.

This soft margin SVM formulation is often called as primal formulation. Using Optimization methods theory we can show that there is an equivalent Dual formulation of this primal formulation. Below is the dual formulation of SVM.

$$\underset{\alpha_i}{max}(\sum_{i=1..n} \alpha_i - \frac{1}{2} \sum_{i=1..n} \sum_{j=1..n} \alpha_i \alpha_j y_i y_j x_i^T x_j)$$

$$\alpha_i \geq 0$$

$$\sum_{i=1..n} \alpha_i y_i = 0$$

Using concepts of optimization theory we can prove that solving primal formulation problem is equivalent to solving dual formulation problem.

Below are few important properties of dual form:

1. For every x_i in primal we have α_i in dual formulation.

2. In dual formulation every x_i only occur in the form of $x_i^T x_j$.

3. Once we solve dual form and find value of α we can get respective value of x for primal form using below equation.

$$\mathrm{f(x}_q) = \sum_{i=1..n} \alpha_i y_i x_i^T x_q + b$$

4. $\alpha_i > 0$ only for support vectors and $\alpha_i = 0$ for non support vectors. So to compute $f(x_q)$ only points that matter are support vectors. This is how we came to name support vector machines as only this support vectors matter for the classification. As all other points than support vectors do not change value of $f(x_q)$.