CRITIQUE

Support-Vector Networks

- Corinna Cortes , Vladimir Vapnik AT&T Bell Labs., Holmdel, NJ 07733, USA

The Support-vector Networks (SVN) added a new dimension to the field of Artificial Intelligence. Although it is fairly recent paper (1995), its implementation spread like wildfire especially in OCR, Cancer detection, Cyber Security, text classification. SVN is a binary classifier. Unlike perceptron and other neural network models this model works by classifying the data using an optimized hyper-plane with the widest margin of separation between the 2 classes. The authors discuss about the different situation that arise while analysing datasets. When the dataset is clearly separable, the hyper-plane can be optimized to fit by solving a quadratic programming problem. But as we all know that datasets are very noisy. In such cases the soft margin hyper plane is optimized to maximize the margin with the least error. But there are also situations where the data points are well mixed and there is no clear gap. The process explained by the author adds dimension to the data points using kernel and then derives the condition which is required to be satisfied in order to uses the kernel. Using this kernel the SVN successfully separates the two classes.

Describing the general features of the SVN, the authors tell us how general the network is. In the end they work on the implementation of SVN on US Postal Database and NIST dataset and compare the performance with the other NN models. Safeguarding against over fitting, faster training time even after increasing the dimensionality proved it to be the ideal choice for such applications.

Optimal Hyperplane:

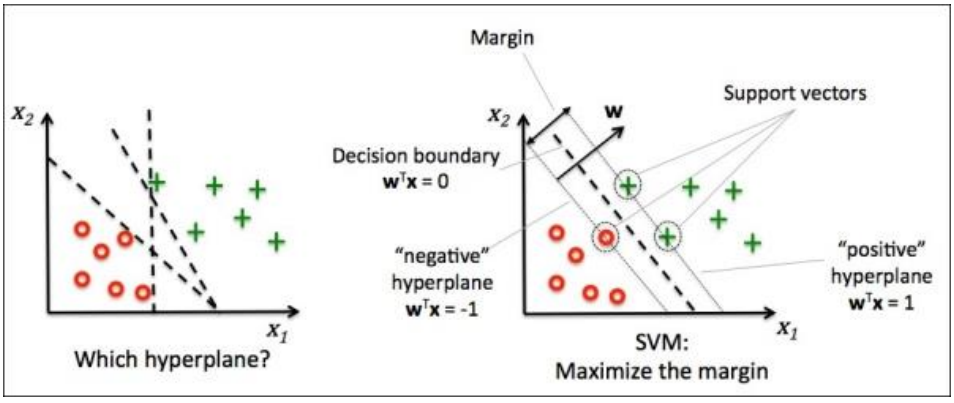


Figure 1

- The authors review the situation where the Data points are separable with a margin without any error.

- Having multiple solutions for the margin of separation the Lagrangian multiplier is used to find the margin with the maximum margin.
- Hence our goal: Maximize the margin $M(\rho_0)$.
- The Author found out required objective equation and constraints for finding the optimal margin as stated below:

$$\therefore \textit{The new goal}: \quad \mathbf{min}\left(\frac{1}{2}\|\vec{w}\|^2\right)$$

$$\& \textit{ the constraints are}: \quad y_i(\vec{w}.\vec{x_i} + b) \geq 1, \forall i = 1, \dots, \dots, n$$

- The Lagrangian of the above equation is:

$$L(w, b, \alpha) = \frac{1}{2}w.w - \sum_{i=1}^{n} \alpha_i.[y_i(w.x_i + b) - 1] \qquad , \alpha_i \geq 0$$

- As the constraint given is an inequality constaint, Kuhn-Tucker condition played an important role in finding the optimum objective functional which is:

$$W(\Lambda) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

$$where,$$

$$\Lambda = [\alpha_i, \dots, \dots, \alpha_n]$$

- The above equation is a quadratic programming problem whose solution can be found in linear time.
- Solving the above equation will determine which of the given vectors in the dataset are the support vectors ($\alpha_i > 0$) and which are not ($\alpha_i = 0$)
- Finally the authors draw a relation between the $W(\Lambda)$ and ρ

$$W(\Lambda_0) = \frac{2}{M^2} = \frac{2}{\rho_0^2}$$

- The derivation is given at the end of the Critique.

Soft margin hyperplane:

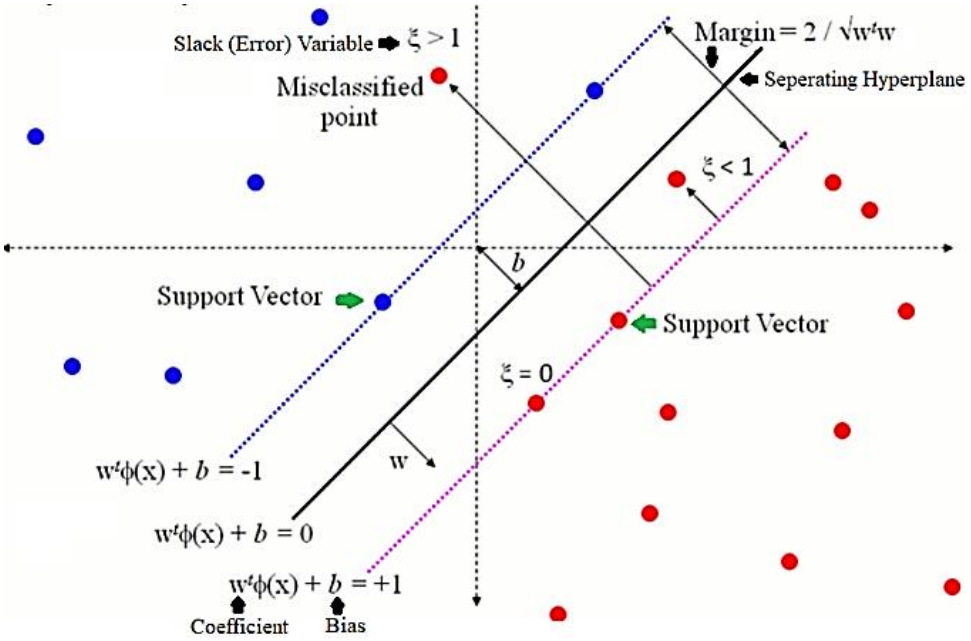
- The major issue with hard margin hyper plane is that the solution is obtained only when the dataset is well separated with each class occupying certain areas in the feature space. But real world data doesn't exist that way. Noisy datasets will prevent the functional form finding the solution.
- The authors introduce ξ_i which is the deviation of the datapoint from their respective margins given in the picture given below. The functional given below is required to be minimized while solving for the optimum margin.

$$\Phi(\xi) = \sum_{i=1}^{\ell} \xi_i^{\sigma}$$

- Modifying the same constaint equation as given in hard margin hyperplane:

for small $\sigma > 0$, subject to the constraints

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, \ell,$$
$$\xi_i \geq 0, \qquad i = 1, \ldots, \ell.$$



- In order to find the optimum margin the functional can be represented as:

$$\frac{1}{2}\mathbf{w}^2 + CF\left(\sum_{i=1}^{\ell} \xi_i^{\sigma}\right)$$

F(u) is monotonic convex function and C is a Regularization Constant

- The Lagrangian of the above equation:

$$L(\mathbf{w}, \xi, b, \Lambda, \mathbf{R})$$
$$= \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C\left(\sum_{i=1}^{\ell} \xi_i\right)^k - \sum_{i=1}^{\ell} \alpha_i [y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i] - \sum_{i=1}^{\ell} r_i \xi_i,$$

- Using the same method of obtaining the solution for hard margin the solution of the above equation is :

$$W(\Lambda, \delta) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j - \frac{\delta^{k/k-1}}{(kC)^{1/k-1}}\left(1 - \frac{1}{k}\right).$$

Where,

$$\delta = \alpha_i + r_i.$$

- The vectorized form of the equation is given as:

$$W(\mathbf{\Lambda}, \delta) = \mathbf{\Lambda}^T \mathbf{1} - \left[\frac{1}{2}\mathbf{\Lambda}^T \mathbf{D}\mathbf{\Lambda} + \frac{\delta^{k/k-1}}{(kC)^{1/k-1}}\left(1 - \frac{1}{k}\right)\right],$$

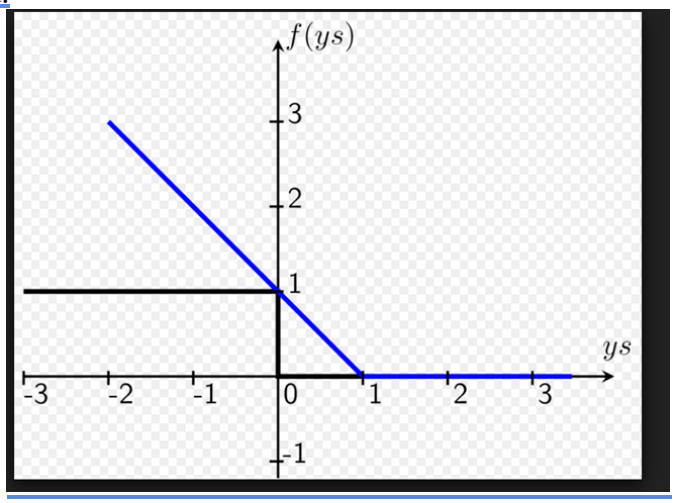
Subjected to the constraints,

$$\mathbf{\Lambda}^T \mathbf{Y} = 0,$$
$$\mathbf{\Lambda} + \mathbf{R} = \delta \mathbf{1},$$
$$\mathbf{\Lambda} \geq \mathbf{0},$$
$$\mathbf{R} \geq \mathbf{0}.$$

- Deducing from the constaint given:

$$\mathbf{0} \leq \mathbf{\Lambda} \leq \delta \mathbf{1}.$$

- If the error distances vary as a Hinge function, Then there can be a higher optimization of the hyperplane because the hyperplanes too close to the data points can be prevented.



- So the author arrives at the solution:

$$\delta = \alpha_{\max} = \max(\alpha_1, \ldots, \alpha_\ell).$$

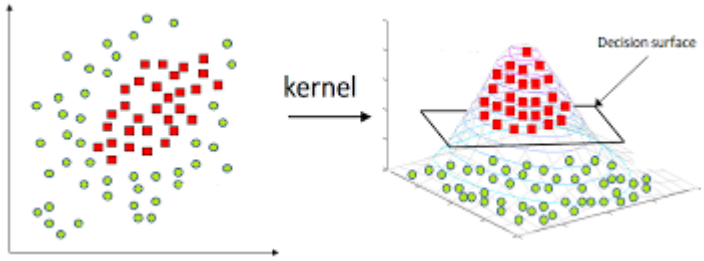
Thus the objective functional

$$W(\mathbf{\Lambda}) = \mathbf{\Lambda}^T \mathbf{1} - \left[\frac{1}{2}\mathbf{\Lambda}^T \mathbf{D}\mathbf{\Lambda} + \frac{\alpha_{\max}^{k/k-1}}{(kC)^{1/k-1}}\left(1 - \frac{1}{k}\right)\right].$$

- In order to find the best $\mathbf{\Lambda}$ one has to solve the above quadratic functional.
- The author recommends to find the optimal soft margin hyper plane by solving the dual quadratic programming problem

Method of convolution of the dot-product in feature Space

- In this section the author gives a solution to form a hyper plane in the feature space instead of input space



- The vectors are first transformed from n dimension to N dimension vector.
- So the classification function can be represented as

$$f(\mathbf{x}) = \phi(\mathbf{x}) \cdot \mathbf{w} + b = \sum_{i=1}^{\ell} y_i \alpha_i \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i) + b.$$

- The Author explains the concept of using Kernel function $K(\mathbf{u}, \mathbf{v})$
- The functions that satisfy Mercer's Theorem can be used to represent the vectors in the feature space.
- The method of determining the decision surface remains same the only difference is that $\mathbf{x}_i \cdot \mathbf{x}_j$ is replace with $\phi(\mathbf{u}) \cdot \phi(\mathbf{v}) \equiv K(\mathbf{u}, \mathbf{v})$.

General features of SVN

- In this section the author explains the benefit of using SVN for classification
- As the solution is obtain just by solving the quadratic functional, the time required to solve it is less and the techniques are very standard
- One can include priori knowledge of the problem or can use any kernel function of choice to determine the hyperplane. Thus this proves SVN to be the most generalized universal machine of it time.
- The author claims that the SVN can be fine tuned according to requirement. As we have seen in the case of soft margin hyperplane the C parameter can be changed to fit the decision surface according to requirement.

Experimental analysis:

- The author implements the SVN in 2 types of data sets, one using the US Postal Service digit database and the other using NIST dataset.
- The kernel used for drawing the hyperplane was:

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$$

For both of the experiments d = 2 was taken.

- For the US postal database preprocessing was done to smooth out the digits using Gaussian method
- The main highlight that we can observe is the that SVN remained immune to overfitting even after increasing the degree of polynomial after 2 as the dimensionality increased 10^10 time but error rate remained same after that.
- In the case of NIST dataset the error rate was comparable to the error rate of the LeNet4 which was the best performing classifier at the time.
- More over the best thing I liked about the SVN was the time it required to train. As claimed by the author the training time was significantly less than the other popular models at that time.

Conclusion

Even though the SVN came fairly recently the application of it has been done in every emerging field namely medical field, cyber security and text & digit recognition. In the era where curse of dimensionality looms around this model uses kernels to increase their dimension and still then find the best fit curve without over fitting it. Although there are better models which can predict with better accuracy but the benefit of using SVN is many fold with less training time, immunity to over fitting and moreover has the wide range of function that can be used as hyperplane.

Derivation of Hard Margin Hyperplane

- We know that the two hyperplanes lie on the Support Vectors and there are no points between them. Hence, we can first consider the mid-hyperplane between these two hyperplanes. The equation of this line would be

$$\boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} = \mathbf{0}$$

And considering that the distance between this line and the hyperplanes is 1, we get

$$\boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} = \mathbf{1}$$

$$\&$$

$$\boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} = -\mathbf{1}$$

It means that there are ***NO*** points where

$$\boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} < \mathbf{1} \,\&\, \boldsymbol{w}.\boldsymbol{x} + \boldsymbol{b} > -\mathbf{1}$$

$$\because \textit{No points lie between the two hyperplanes}$$

$$i.e., \boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \geq \mathbf{1} \; \boldsymbol{or} \; \boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \leq -\mathbf{1}, \forall \; \boldsymbol{i} = \mathbf{1}, \ldots\ldots, \boldsymbol{n}$$

For points where,

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \geq \mathbf{1}, \boldsymbol{y_1} = \mathbf{1}$$

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \leq -\mathbf{1}, \boldsymbol{y_i} = -\mathbf{1}$$

- So here are our constraints combined after multiplying with y_i for convinience:

$$\boldsymbol{y_i}(\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b}) \geq \mathbf{1}, \forall \; \boldsymbol{i} = \mathbf{1}, \ldots, \ldots, \boldsymbol{n} \quad -(\mathbf{1})$$

Now that we have the equations of the hyperplane, it makes sense to define the *Margin* M in terms of the equation or to be specific, in terms of the variables $\vec{w}$ *and* b.

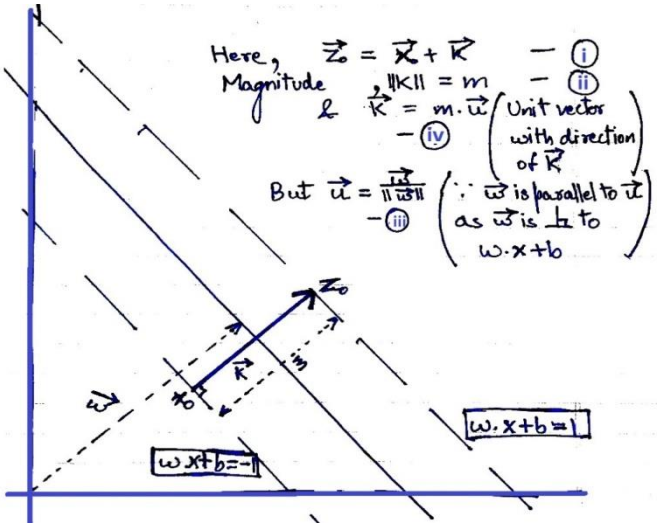


Figure 2

- In the above *fig.2* $\vec{k}$ is the distance vetor between the hyperplanes and has the direction of the normal vector $\vec{w}$ and magnitude m.

$\because \vec{z_0}$ lies on $\vec{w}.\vec{x}+b=1$

$\Rightarrow \quad \vec{w}.\vec{z_0}+b=1$

$\Rightarrow \quad \vec{w}.(\vec{x_0}+\vec{k})+b=1$

$\Rightarrow \quad \vec{w}.\vec{x_0}+\vec{w}.\vec{k}+b=1 \qquad -(V)$

But $\quad \vec{w}.\vec{x_0}+b=-1 \quad \because \vec{x_0}$ lies on it.

& $\quad \vec{k}=m.\vec{u}=m.\frac{\vec{w}}{\|\vec{w}\|}$

Substituting these in (V),

$\Rightarrow \quad -1+m.\frac{\vec{w}}{\|\vec{w}\|}.\vec{w}=1$

$\Rightarrow \quad m.\frac{\|\vec{w}\|^2}{\|\vec{w}\|}=2$

$\Rightarrow \quad m=\frac{2}{\|\vec{w}\|}=f_0=M \qquad -(2)$

- Now that we have M, we need to maximize it i.e.,

$$\max(M)=\max\left(\frac{2}{\|\vec{w}\|}\right)=\min(\|\vec{w}\|)$$
$$=\min\left(\frac{1}{2}\|\vec{w}\|\right) \qquad [\textit{For mathematical convinience}]$$

$\therefore$ *The new goal*: $\min\left(\frac{1}{2}\|\vec{w}\|^2\right) \qquad -(3)$

& *the constraints are*: $y_i(\vec{w}.\vec{x_i}+b)\geq 1, \forall i=1,\dots,\dots,n \qquad -(4)$

- The above equations are **non – linear constraint equations**. Generally an objective function with equality constraint can be solved using Lagrangian multiplier but here

there is an inequality constraint. So the Kuhn- Tucker Condition needs to be applied inorder to find the optimal solution.

The Lagrangian eq.:

$$L(w, b, \alpha) = \frac{1}{2} w.w - \sum_{i=1}^{n} \alpha_i . [y_i (w.x_i + b) - 1] \quad , \alpha_i \geq 0 \qquad -(6)$$

$\alpha_i \geq 0$ is one of the necessary KKT conditions as we will see later.

Just like we find the derivative of an equation and equate it to 0 for an optimal solution (maxima or minima) for unconstrained equations, here too we compute the derivatives and equate them to 0.

$$\therefore \frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i . y_i . x_i = 0$$

$$\Rightarrow w = \sum_{i=1}^{n} \alpha_i . y_i . x_i \qquad -(7)$$

$$\& \quad \frac{\partial L}{\partial b} = -\sum_{i=1}^{n} \alpha_i . y_i = 0 \qquad -(8)$$

Substituting these back into (6) and deducing, we arrive at an equation which is a function of Λ ($\Lambda = [\alpha_i, \dots, \dots, \alpha_n]$) called the Objective function i.e.,

$$W(\Lambda) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \qquad -(9)$$

- Now that we have this objective functional, we need to optimize the value of Λ for obtaining the minima of (3). From the above $eq.$ it is clear that we are now solving/optimizing a different problem from the original one!! This is popularly called duality. For this duality to hold true (which is another condition of KKT) i.e., to obtain $w_0(optimal\ w)$ from $\Lambda_0(optimal\ \Lambda)$, the following conditions must hold true:
  - **Dual Feasibility:**
    $$\alpha_i \geq 0, \forall\ i = 1, \dots, \dots, n$$
  - **Complementary Feasiblity:**
    $$\alpha_i . [y_i (w.x_i + b) - 1] = 0 \qquad -(10)$$
  - **Stationary(flat) Condition:** Given by equations of the form $eqs.(7)$ & (8)
  - **Primal Feasibility:**
    Our original constraint, $y_i (w.x_i + b) - 1 \geq 0, \forall i = 1, \dots, \dots, n$

- Lastly, we must understand the concept of Duality and Complementary Slackness. It must first be observed that our primal problem is a convex problem. Now what is duality?

- Ans. When a problem can be converted to another problem whose solution is easier to compute and also provides the solution for the original problem, the two problems are said to exhibit Duality and the vice-versa holds true. This is called complementary slackness. But all dual functions need not necessarily have a solution providing the optimal value for the other. This can be inferred from the below *Fig. 3* where there is a Duality Gap between the primal and the dual problem. In the *Fig. 4*, the dual problems exhibit strong duality and have complementary slackness. Also, it is clear from this graph that a minimization problem is converted to a maximization one. Hence we need to maximize α_i.

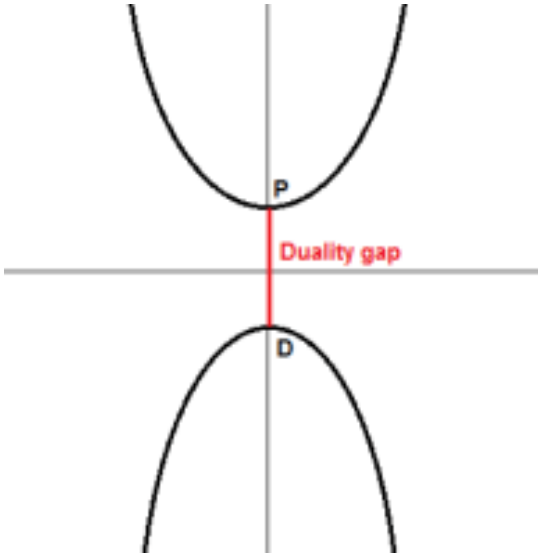


Figure 3: Weak Duality due to the Duality Gap.

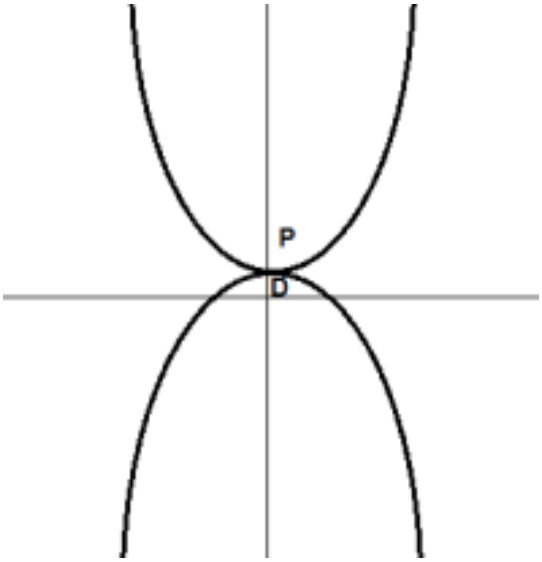


Figure 4: Strong Duality due to no Duality Gap (Exhibits Complementary Slackness)

- Now consider $eq.(10)$, here either $\alpha_i = 0$ *or* $[y_i(w.x_i + b) - 1] = 0$. When the 2nd occurs, $\alpha_i \neq 0 \Rightarrow \alpha_i > 0$ and the vectors or data points lie on the Maximum Margin Hyperplanes.

$$\therefore \ \boldsymbol{\alpha_i > 0 \ for \ support \ vectors}$$

Directly representing the margin M in terms of α_0 $(Max.\alpha)$

From $eq.(7)$ *W.K.T,*

$$\boldsymbol{w_0 = \sum_{i=1}^{n} \alpha_i^0 . y_i . x_i}$$

$$\boldsymbol{\Rightarrow w_0 . w_0 = \sum_{i=1}^{n} \alpha_i^0 . y_i . x_i . w_0}$$

$$= \sum_{i=1}^{n} \alpha_i^0 . (1 - y_i . b_0)$$

$$\Rightarrow w_0 . w_0 = \sum_{i=1}^{n} \alpha_i^0 \qquad -(11)$$

From the objective functional,

$$W(\Lambda_0) = \sum_{i=1}^{n} \alpha_i^0 - \frac{1}{2} w_0 . w_0$$

$$= \frac{1}{2} w_0 . w_0$$

$$\Rightarrow W(\Lambda_0) = \frac{2}{M^2} = \frac{2}{\rho_0^2} \qquad [Substituting\ for\ w_0\ from\ (2)]$$

Useful links:

1. https://www.svm-tutorial.com/2014/11/svm-understanding-math-part-1/
2. https://www.svm-tutorial.com/2014/11/svm-understanding-math-part-2/
3. https://www.svm-tutorial.com/2015/06/svm-understanding-math-part-3/
4. https://www.svm-tutorial.com/2016/09/unconstrained-minimization/
5. https://www.svm-tutorial.com/2016/09/convex-functions/
6. https://www.svm-tutorial.com/2016/09/duality-lagrange-multipliers/
7. Support Vector Machines Succinctly by Alexandre Kowalczyk
8. Ali Ghodsi, Lec 12: Soft margin Support Vector Machine (svm)
9. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5822181/
10. https://www.youtube.com/watch?v=JTTiELgMyuM
11. http://fourier.eng.hmc.edu/e161/lectures/svm/
12. http://www.svcl.ucsd.edu/courses/ece271B-F09/handouts/SoftSVMs.pdf
13. Lecture 70 — Soft Margin SVMs | Mining of Massive Datasets | Stanford University