Support Vector Machines

MFML Team

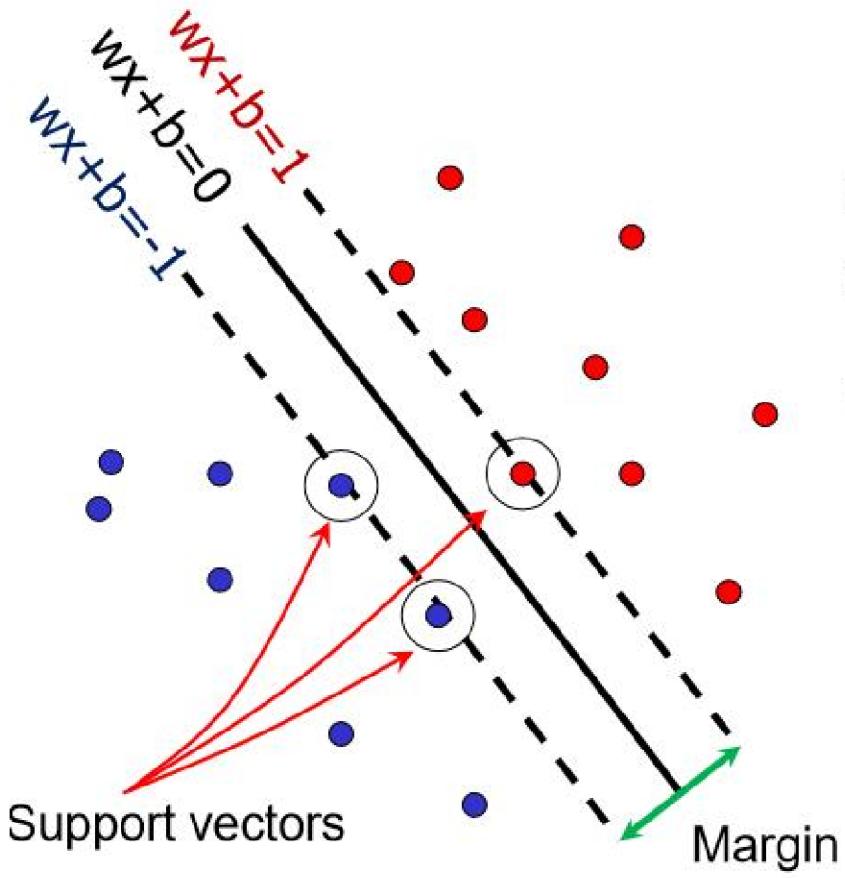
BITS Pilani
Pilani Campus

Topics to be covered

- Linear Classifiers
- Maximum Margin Classification
- Linear SVM
- SVM optimization problem
- Soft Margin SVM

Support Vector Machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

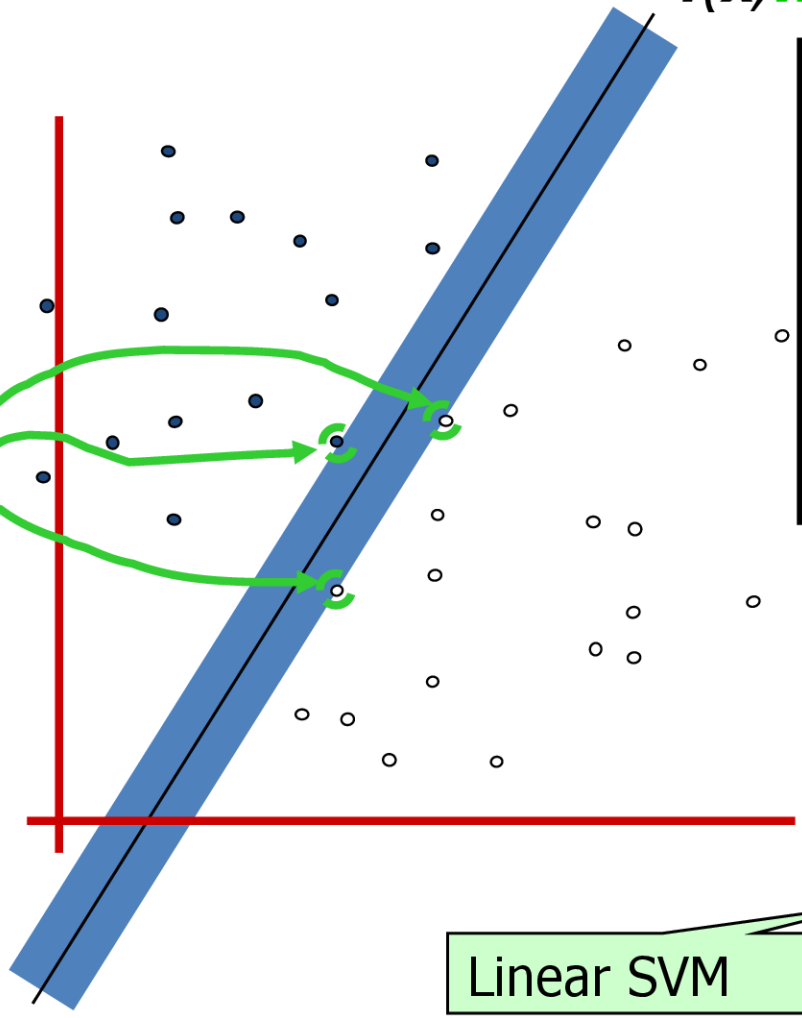
For support vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Maximum Margin

$$f(x, w, b) = sign(w \cdot x + b)$$

denotes +1

denotes -1



Support Vectors are those datapoints that the margin pushes up against

1. If hyperplane is oriented such that it is close to some of the points in your training set, new data may lie on the wrong side of the hyperplane, even if the new points lie close to training examples of the correct class.
2. Solution is maximizing the margin

The maximum margin linear classifier is the linear classifier with the, maximum margin.

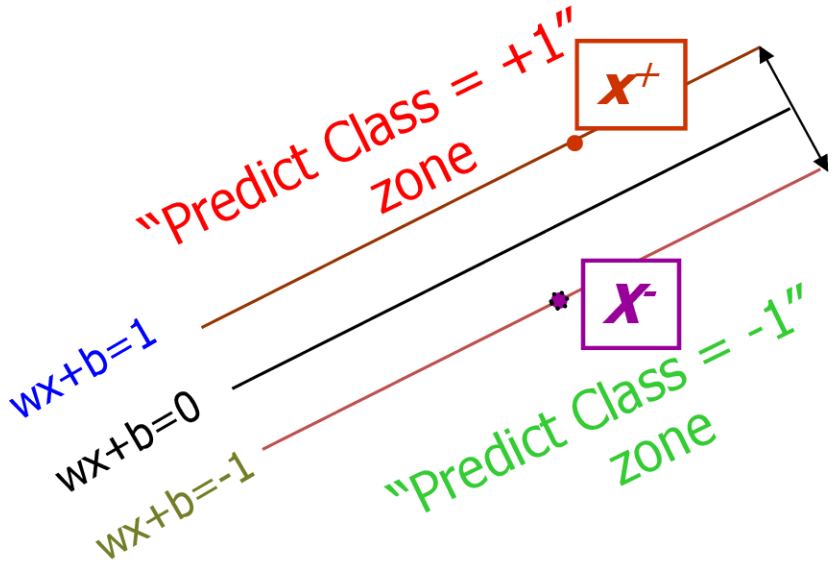
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

Support Vectors

- Geometric description of SVM is that the max-margin hyperplane is completely determined by those points that lie nearest to it.
- Points that lie on this margin are the support vectors.
- The points of our data set which if removed, would alter the position of the dividing hyperplane

Linear SVM Mathematically



M=Margin Width

What we know:

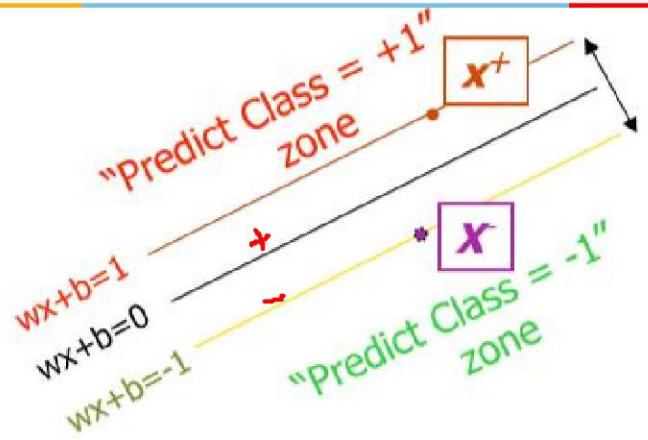
- $\boldsymbol{w} \cdot \boldsymbol{x}^{+} + b = +1$
- $\boldsymbol{w} \cdot \boldsymbol{x}^{-} + b = -1$

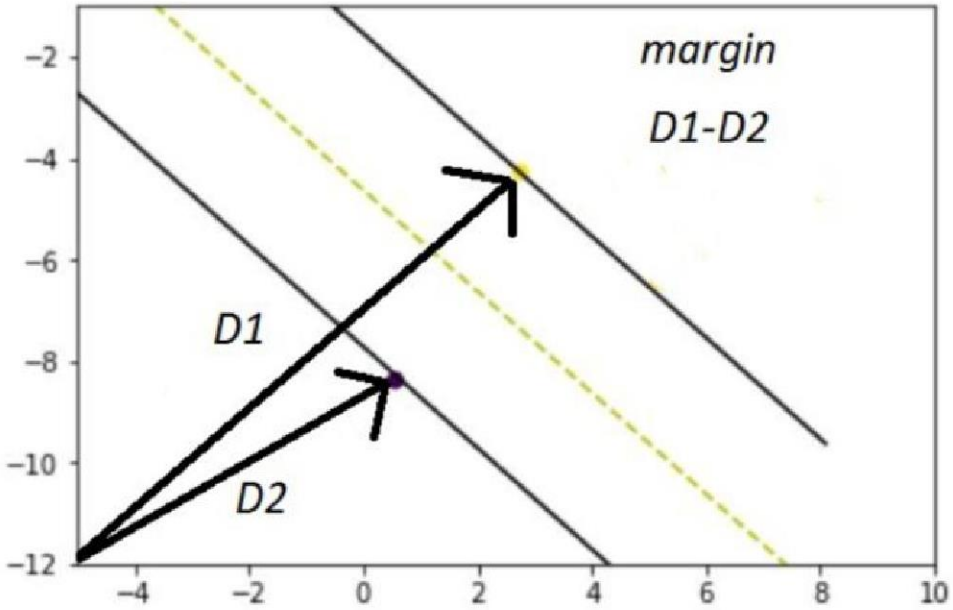
Distance between lines given by solving linear equation:

Maximize margin: $M = \frac{2}{||w||}$

Equivalent to minimize: $\frac{1}{2}||w||^2$

Linear SVM Mathematically





$D1 = w^T x+b=1$ $\quad w^T x+b-1=0$

$D2= w^T x+b=-1$ $\quad w^T x+b+1=0$

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

⇩ Solve algebraically

$$\frac{2}{|w|}$$

Optimization Problem

1. Maximize margin $2/||\mathbf{w}||$
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

Quadratic optimization problem:

Find $\mathbf{w}$ and b such that

$\mathbf{\Phi(w)} = \frac{1}{2}||\mathbf{w}||^2$ is minimized;

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\,(\mathbf{w^T x_i} + b) \geq 1$

$y_i\,(\mathbf{w^T x_i} + b) \geq 1$

$+1(\mathbf{w^T x_i} + b) \geq 1$

$-1(\mathbf{w^T x_i} + b) \leq 1$

same as $(\mathbf{w^T x_i} + b) \geq 1$

Solving the Optimization Problem

Find $\mathbf{w}$ and b such that

$\Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$ is minimized; Type equation here.

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

← Primal

- **Need to optimize a *quadratic* function subject to *linear inequality* constraints.**
- **All constraints in SVM are linear**
- **Quadratic optimization problems are a well-known class of mathematical programming problems, and many (rather intricate) algorithms exist for solving them.**
- **The solution involves constructing a *unconstrained problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:**

Recall : Karush–Kuhn–Tucker (KKT) theorem

- max f(x) subject to $g_i(x) = 0$ and $h_j(x) \geq 0$ for all i, j
- Make the Lagrangian function

$$\mathcal{L} = f(x) - \sum_i \lambda_i g_i(x) - \sum_j \mu_j h_j(x)$$

- Necessary conditions to have a minimum are

$$\begin{aligned} \nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) &= 0 \\ g_i(x^*) &= 0 \text{ for all } i \\ h_j(x^*) &\geq 0 \text{ for all } j \\ \mu_j &\geq 0 \text{ for all } j \\ \mu_j^* h_j(x^*) &= 0 \text{ for all } j \end{aligned}$$

Solving the Optimization Problem

- **The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:**

$$L(w, b, \alpha_i) = \frac{1}{2}||\mathbf{w}||^2 - \Sigma\, \alpha_i\, [y_i\, (\mathbf{w}^T\mathbf{x_i} + b) - 1]$$

- **Taking partial derivative with respect to w , $\frac{\partial L}{\partial w} = 0$**
  - $w - \Sigma\, \alpha_i y_i\, \mathbf{x_i} = 0$
  - $w = \Sigma\, \alpha_i y_i\, \mathbf{x_i}$
- **Taking partial derivative with respect to b, $\frac{\partial L}{\partial b} = 0$**
  - $-\Sigma\, \alpha_i\, y_i = 0$
  - $\Sigma\, \alpha_i\, y_i = 0$

Solving the Optimization Problem

$$L(w, b, \alpha_i) = \frac{1}{2}||\mathbf{w}||^2 - \Sigma\, \alpha_i [y_i (\mathbf{w} \cdot \mathbf{x_i} + b) - 1]$$

- Expanding above equation:

$$L(w, b, \alpha_i) = \frac{1}{2}||\mathbf{w}||^2 - \Sigma\, \alpha_i y_i\, \mathbf{w} \cdot \mathbf{x_i} - \Sigma\, \alpha_i y_i\, b + \Sigma\, \alpha_i$$

- Substituting $w = \Sigma\, \alpha_i y_i\, \mathbf{x_i}$ and $\Sigma\, \alpha_i y_i = 0$ in above equation

$$L(w, b, \alpha_i) = \frac{1}{2} (\Sigma_i\, \alpha_i y_i\, \mathbf{x_i})(\Sigma_j\, \alpha_j y_j\, \mathbf{x_j}) - (\Sigma_i\, \alpha_i y_i\, \mathbf{x_i})(\Sigma_j\, \alpha_j y_j\, \mathbf{x_j}) + \Sigma\, \alpha_i$$

$$L(w, b, \alpha_i) = \Sigma\, \alpha_i - \frac{1}{2} (\Sigma_i\, \alpha_i y_i\, \mathbf{x_i})(\Sigma_j\, \alpha_j y_j\, \mathbf{x_j})$$

$$L(w, b, \alpha_i) = \Sigma\, \alpha_i - \frac{1}{2} (\Sigma_i \Sigma_j\, \alpha_i\, \alpha_j y_i\, y_j\, \mathbf{x_i} \cdot \mathbf{x_j})$$

Support Vectors

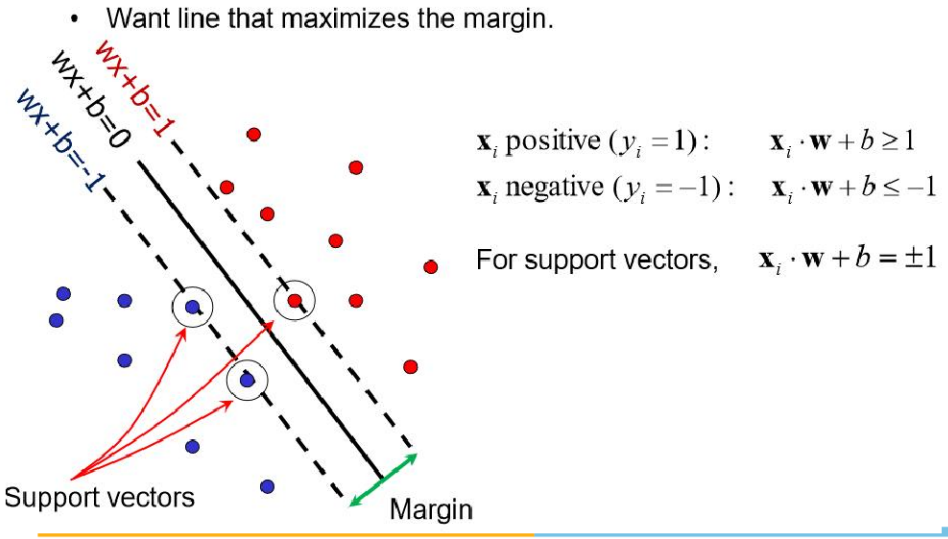
Using KKT conditions : $\alpha_i [y_i (\mathbf{w}^T\mathbf{x_i} + b) -1]=0$

For this condition to be satisfied either $\alpha_i =0$ and $y_i (\mathbf{w}^T\mathbf{x_i} + b) -1 > 0$
OR $y_i (\mathbf{w}^T\mathbf{x_i} + b) -1=0$ and $\alpha_i > 0$

For support vectors:$y_i (\mathbf{w}^T\mathbf{x_i} + b) -1=0$

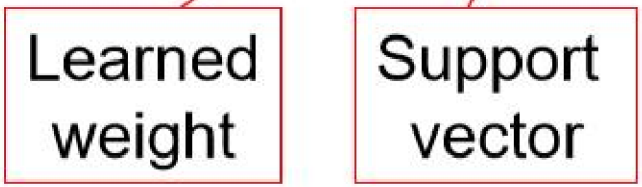
$$L(w, b, \alpha_i)= \frac{1}{2}||\mathbf{w}||^2 - \Sigma \alpha_i [y_i (\mathbf{w}^T\mathbf{x_i} + b) -1]$$

For all points other than support vectors:$\alpha_i =0$



Solving the Optimization Problem

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$



Solving the Optimization Problem

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

- Classification function:

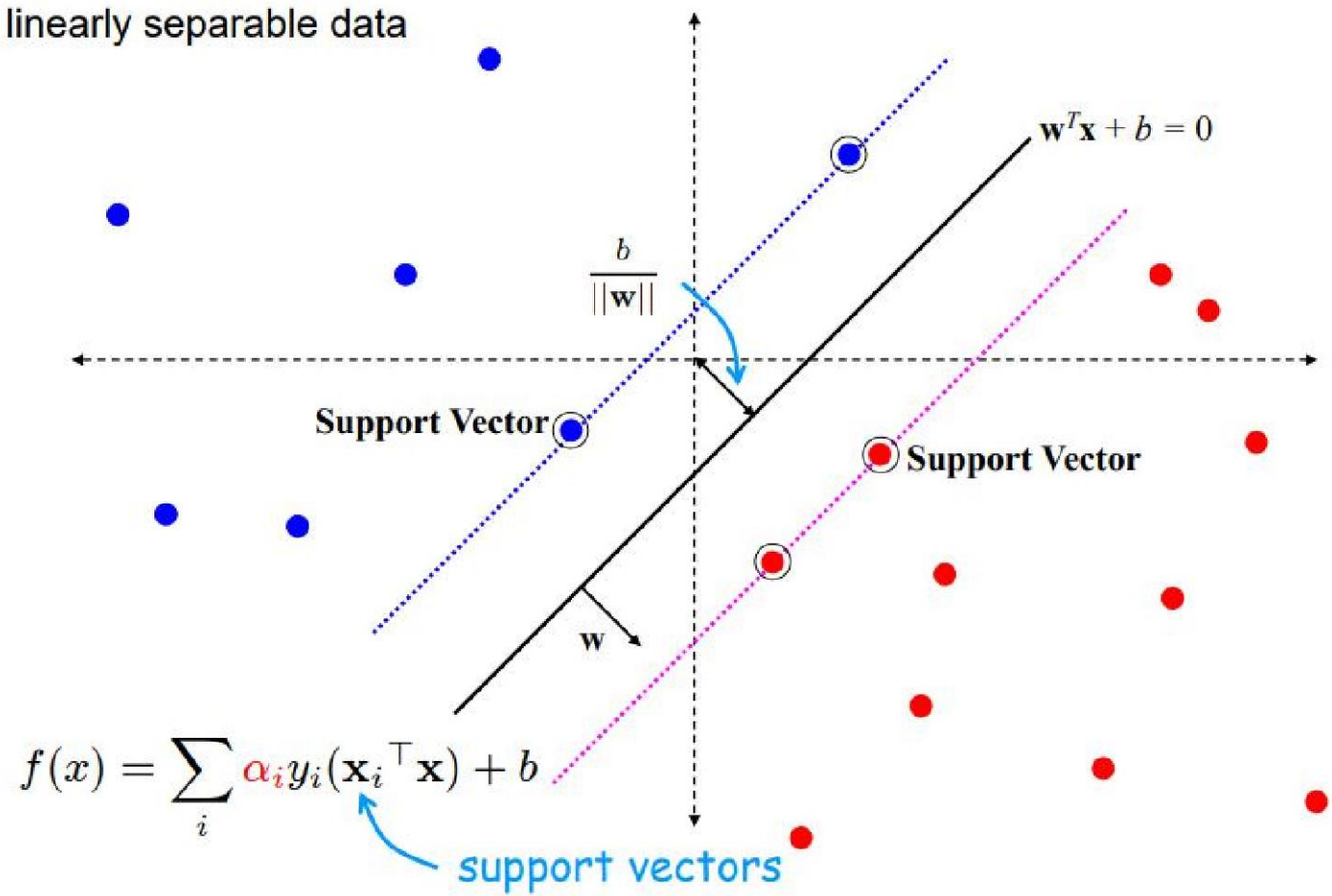
$$f(x) = \text{sign}\,(\mathbf{w} \cdot \mathbf{x} + \text{b})$$

$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

If f(x) < 0, classify as negative, otherwise classify as positive.

- Notice that it relies on an *inner product* between the test point ***x*** and the support vectors $\boldsymbol{x}_i$
- (Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points)

Substituting w in support vectors function



Linear SVM: Numerical Problem

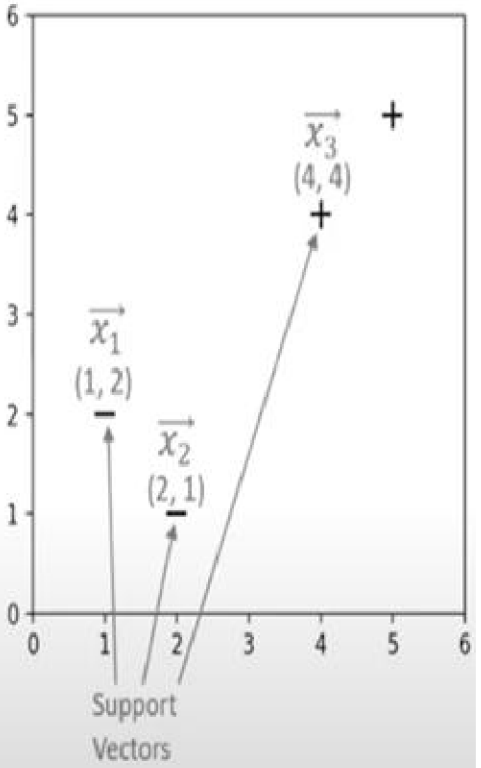
$$\sum_i \alpha_i y_i \vec{x_i} \cdot \vec{x} + b = 1 \quad \text{if } \vec{x} \text{ is + ve support vector}$$

$$\sum_i \alpha_i y_i \vec{x_i} \cdot \vec{x} + b = -1 \quad \text{if } \vec{x} \text{ is - ve support vector}$$

$$\sum_i \alpha_i y_i = 0$$

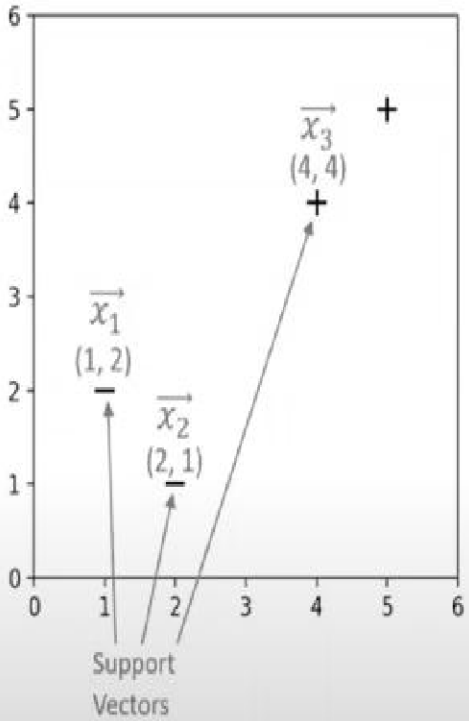
Let $\beta_i = \alpha_i y_i$

$$\begin{cases} \beta_1 \vec{x_1} \cdot \vec{x_1} + \beta_2 \vec{x_2} \cdot \vec{x_1} + \beta_3 \vec{x_3} \cdot \vec{x_1} + b = -1 \\ \beta_1 \vec{x_1} \cdot \vec{x_2} + \beta_2 \vec{x_2} \cdot \vec{x_2} + \beta_3 \vec{x_3} \cdot \vec{x_2} + b = -1 \\ \beta_1 \vec{x_1} \cdot \vec{x_3} + \beta_2 \vec{x_2} \cdot \vec{x_3} + \beta_3 \vec{x_3} \cdot \vec{x_3} + b = 1 \\ \beta_1 + \beta_2 + \beta_3 = 0 \end{cases}$$



Numerical: Linear SVM

$$\begin{cases} \beta_1 \vec{x_1} \cdot \vec{x_1} + \beta_2 \vec{x_2} \cdot \vec{x_1} + \beta_3 \vec{x_3} \cdot \vec{x_1} + b = -1 \\ \beta_1 \vec{x_1} \cdot \vec{x_2} + \beta_2 \vec{x_2} \cdot \vec{x_2} + \beta_3 \vec{x_3} \cdot \vec{x_2} + b = -1 \\ \beta_1 \vec{x_1} \cdot \vec{x_3} + \beta_2 \vec{x_2} \cdot \vec{x_3} + \beta_3 \vec{x_3} \cdot \vec{x_3} + b = 1 \\ \beta_1 + \beta_2 + \beta_3 = 0 \end{cases}$$



Plug in the values of all the vectors

$$\begin{cases} 5\beta_1 + 4\beta_2 + 12\beta_3 + b = -1 \\ 4\beta_1 + 5\beta_2 + 12\beta_3 + b = -1 \\ 12\beta_1 + 12\beta_2 + 32\beta_3 + b = 1 \\ \beta_1 + \beta_2 + \beta_3 = 0 \end{cases}$$

$$\Rightarrow \begin{cases} \beta_1 = -0.08 \\ \beta_2 = -0.08 \\ \beta_3 = 0.16 \\ b = -2.2 \end{cases}$$

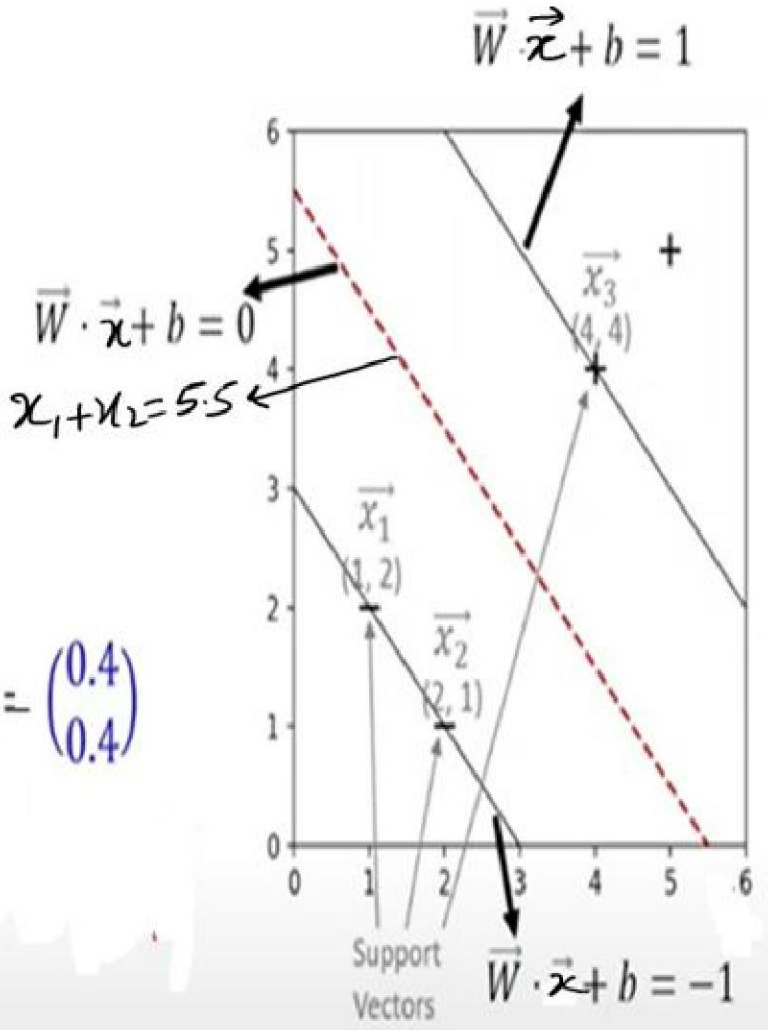
$$\begin{cases} \beta_1 = -0.08 \\ \beta_2 = -0.08 \\ \beta_3 = 0.16 \end{cases}$$

So, hyperplane is essentially those $\vec{x}$ that satisfy the equality.

$$\vec{W} = \sum_i \alpha_i y_i \vec{x}_i \qquad \sum_i \beta_i \vec{x}_i = -0.08\begin{pmatrix}1\\2\end{pmatrix} - 0.08\begin{pmatrix}2\\1\end{pmatrix} + 0.16\begin{pmatrix}4\\4\end{pmatrix} = \begin{pmatrix}0.4\\0.4\end{pmatrix}$$

$$: \vec{W} \cdot \vec{x} + b = 0$$

$$\text{So, } \begin{pmatrix}0.4\\0.4\end{pmatrix} \cdot \begin{pmatrix}x_1\\x_2\end{pmatrix} - 2.2 = 0 \Rightarrow x_1 + x_2 = 5.5$$



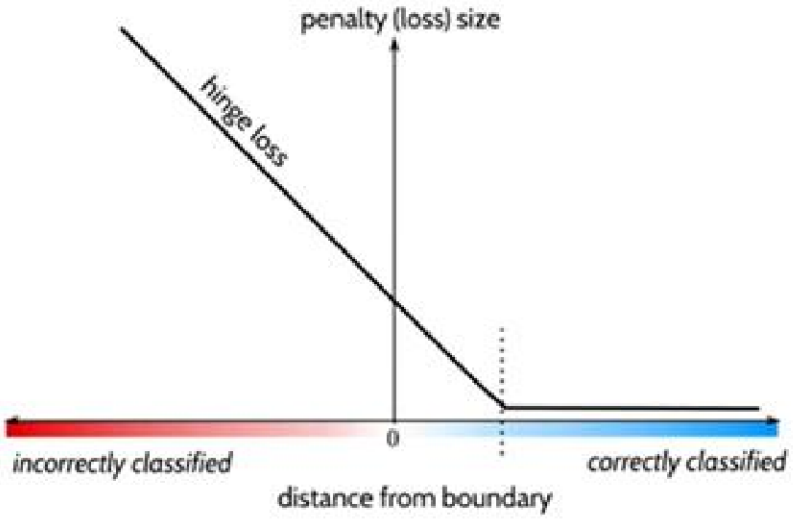
Hinge Loss

Hinge Loss is one of the types of Loss Function, mainly used for maximum margin classification models.

Hinge Loss incorporates a margin or distance from the classification boundary into the loss calculation. Even if new observations are classified correctly, they can incur a penalty if the margin from the decision boundary is not large enough.

$$L = \max(0, 1 - y_i (w^T x_i + b))$$

0 - for correct classification
1 - for wrong classification



Hinge loss

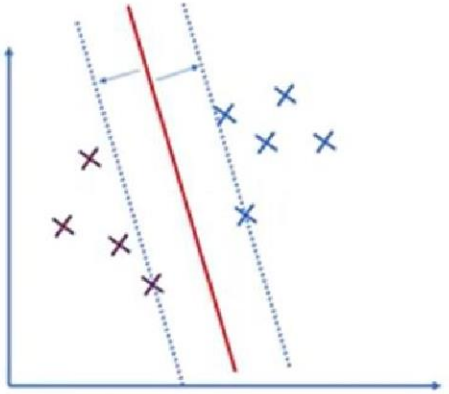
HINGE LOSS – Numerical Example 1

$y' = w^T x + b$

Correctly classification

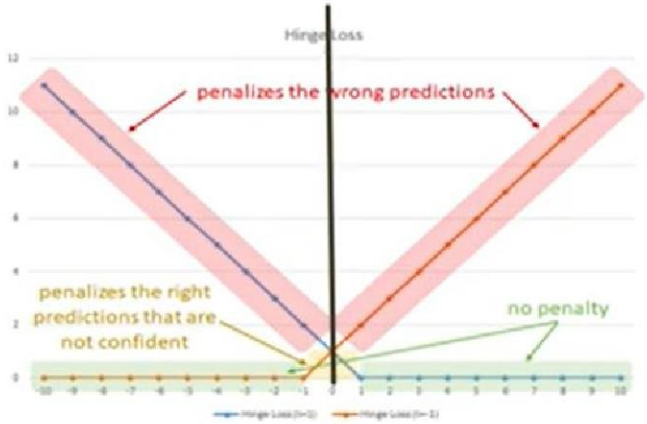
- Actual output $y = +1$. Predicted output $y' = 0.5$.
- Hinge Loss $= max(0,\ 1 - y * y')$
  $= max(0,\ 1 - 1 * 0.5)$
  $= 0.5.$

Hinge loss is close to zero for correctly classified sample.



Misclassification

- Actual output $y = -1$. Predicted output $y' = 0.5$.
- Hinge Loss $= max(0,\ 1 - y * y')$
  $= max(0,\ 1 - (-1) * 0.5)$
  $= 1.5.$
- Hinge loss is higher for misclassified sample.



SVM : Estimating w and b

n : no. of data points

- **Want to minimize $J(w,b)$:**

$$J(w,b) = \frac{1}{2}\sum_{j=1}^{d}\left(w^{(j)}\right)^2 + C\sum_{i=1}^{n}\max\left\{0, 1 - y_i\left(\sum_{j=1}^{d} w^{(j)} x_i^{(j)} + b\right)\right\}$$

Empirical loss $L(x_i\, y_i)$

- **Compute the gradient $\nabla(j)$ w.r.t. $w^{(j)}$**

$$\frac{\partial J(w,b)}{\partial w^{(j)}} = w^{(j)} + C\sum_{i=1}^{n}\frac{\partial L(x_i, y_i)}{\partial w^{(j)}}$$

$$\frac{\partial J}{\partial b} = C\sum_{i=1}^{n}\frac{\partial L(x_i, y_i)}{\partial b}$$

$$\frac{\partial L(x_i, y_i)}{\partial b} = 0 \text{ if } y_i(w x_i + b) \geq 1$$

$$= -y_i \text{ otherwise}$$

$$\frac{\partial L(x_i, y_i)}{\partial w^{(j)}} = 0 \quad \text{if } y_i(w \cdot x_i + b) \geq 1$$

$$= -y_i x_i^{(j)} \quad \text{else}$$

Gradient Descent

Iterate until Convergence

$$Compute: \frac{\partial J}{\partial b} = C \sum_{1}^{n} \frac{\partial L(x_i, y_i)}{\partial b}$$

$$Update: b_{new} \leftarrow b - \eta \frac{\partial J}{\partial b},$$

For j=1…..d

$$Compute: \frac{\partial J}{\partial w^j} = w^j + C \sum_{1}^{n} \frac{\partial L(x_i, y_i)}{\partial w^j}$$

$$Update: w^j_{new} \leftarrow w^j - \eta \frac{\partial J}{\partial w^j},$$

$$b \leftarrow b_{new}, w \leftarrow w_{new}$$

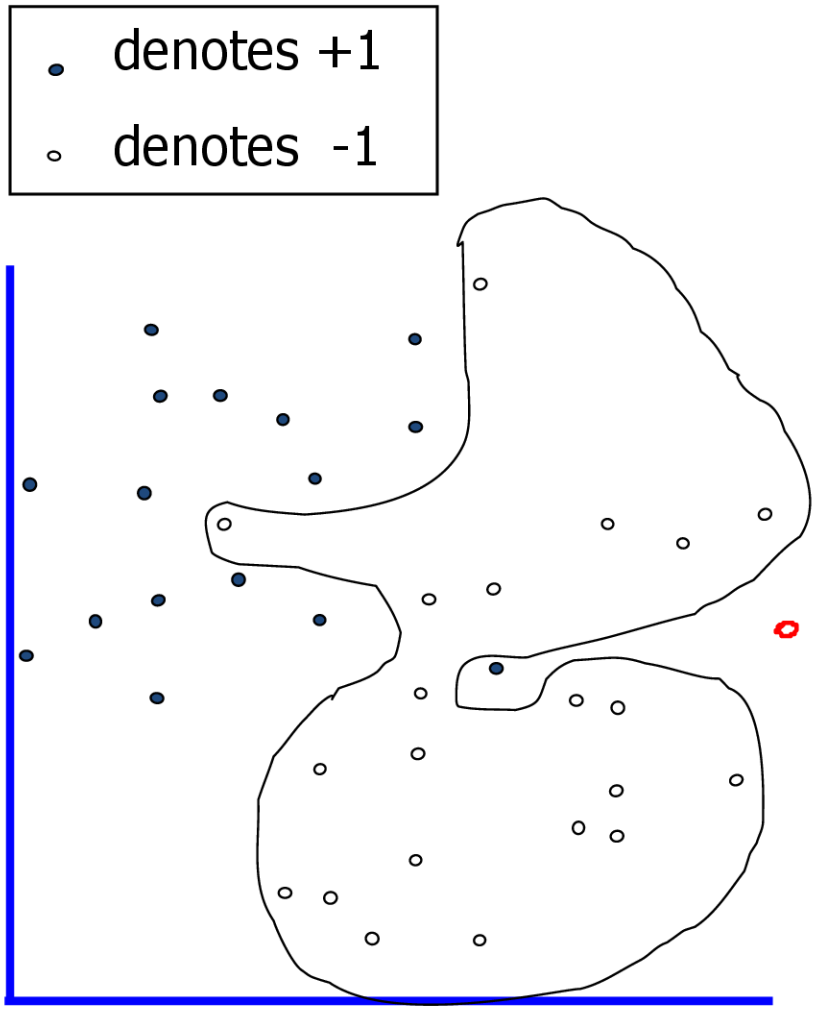
C parameter in cost function

C parameter tells the SVM optimization how much you want to avoid misclassifying each training example.

For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly.

Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points

Dataset with noise



- **Hard Margin: So far we require all data points be classified correctly**

  - No training error

- **What if the training set is noisy?**

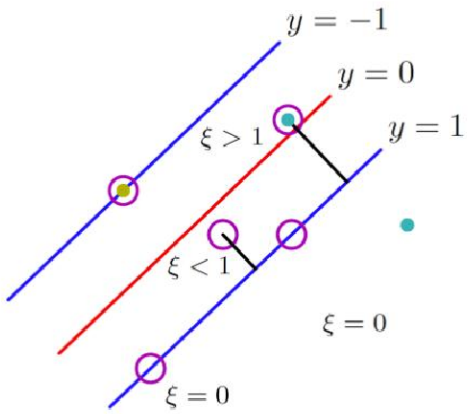
Soft Margin Classification

Slack variables ξ_i **can be added to allow misclassification of difficult or noisy examples.**

What should our quadratic optimization criterion be?

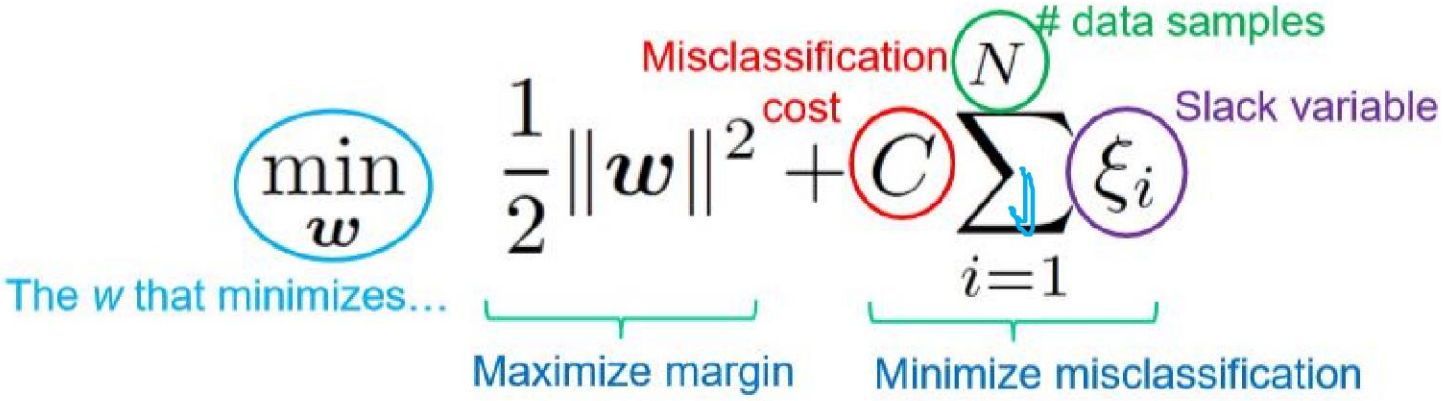
Minimize

$$\frac{1}{2}\mathbf{w}.\mathbf{w} + C\sum_{k=1}^{R}\varepsilon_k$$



- **Slack variable** as giving the classifier some leniency when it comes to moving around points near the **margin**.
- When C is large, larger slacks penalize the objective function of SVM's more than when C is small.

Soft Margin



$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i,$$
$$\xi_i \geq 0, \quad \forall i = 1, \ldots, N$$

Hard Margin versus Soft Margin

- **Hard Margin:**

Find $\mathbf{w}$ and b such that

$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

- **Soft Margin incorporating slack variables:**

Find $\mathbf{w}$ and b such that

$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum \xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

- **Parameter *C* can be viewed as a way to control overfitting.**