



BITS Pilani
Pilani Campus

Machine Learning DSECL ZG565

Support Vector Machines

- Dr. Monali Mavani

Topics to be covered



- Maximum Margin Classification
- Linear SVM
- SVM optimization problem

Support Vector Machines - SVM



- This technique has its roots in statistical learning theory (Vladimir Vapnik, 1992).
- As a task of classification, it searches for optimal hyperplane(i.e., decision boundary) separating the tuples of one class from another.
- SVM works well with higher dimensional data and thus avoids dimensionality problem.

Key Concepts

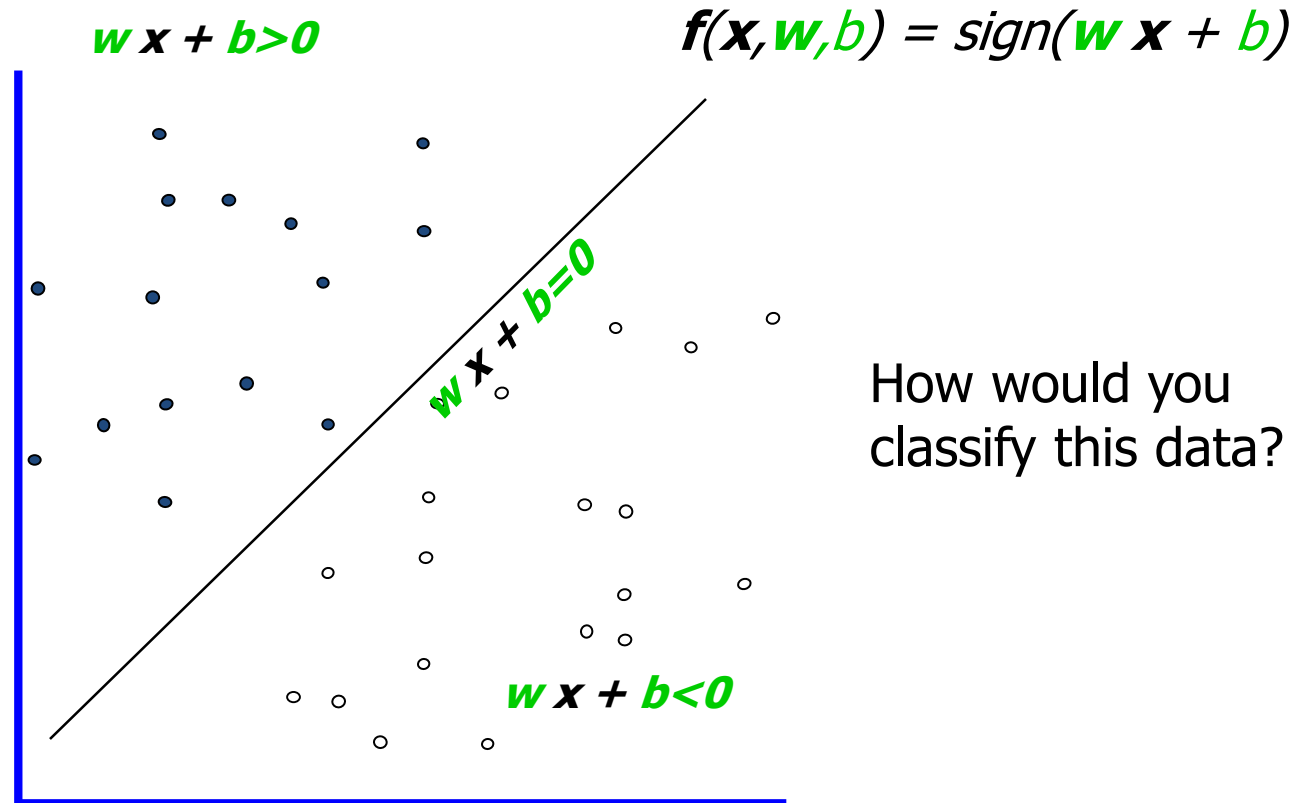


- Maximum margin hyperplane : a key concept in SVM.
- Linear SVM : a classification technique when training data are linearly separable.
 - **Optimization using Langrangian**
- Non-linear SVM : a classification technique when training data are linearly non-separable.
 - **Kernel Trick**
 - Linear SVM with soft margin
 - Non-linear SVM

Maximum Margin Hyperplane

Linear Classifiers

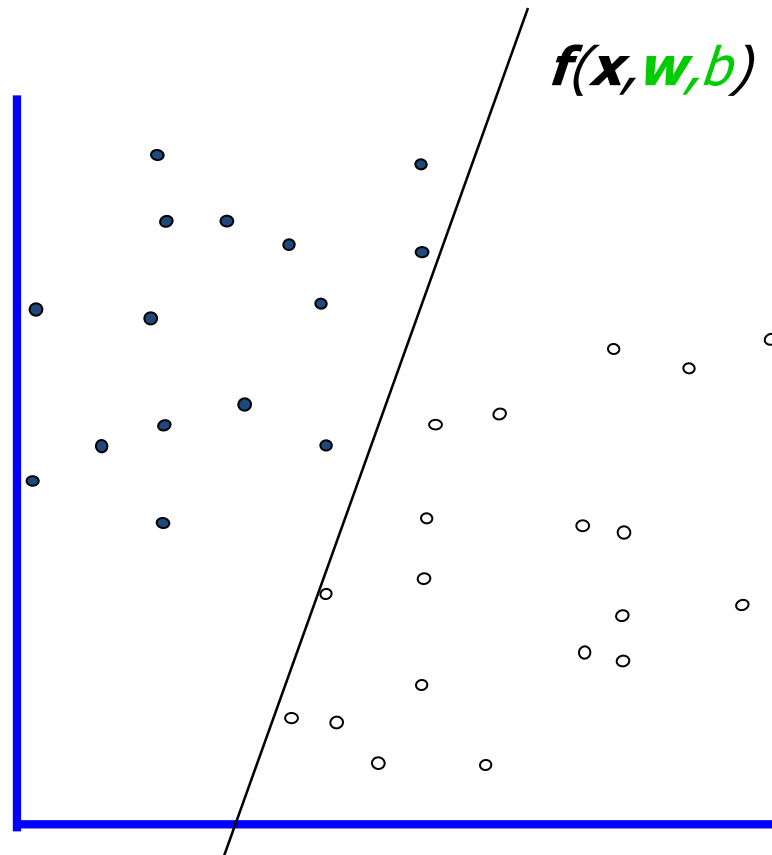
- denotes +1
- denotes -1



How would you classify this data?

Linear Classifiers

- denotes +1
- denotes -1



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

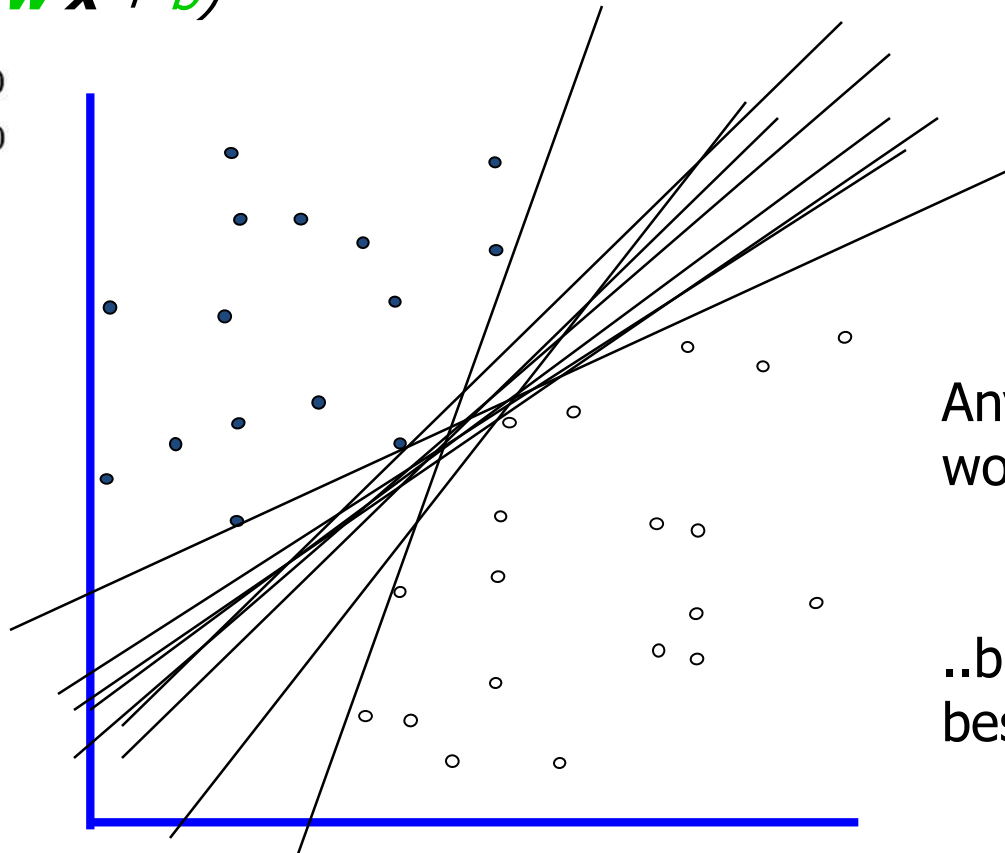
How would you classify this data?

Linear Classifiers

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

$$\mathbf{x}_i \text{ positive : } \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative : } \mathbf{x}_i \cdot \mathbf{w} + b < 0$$



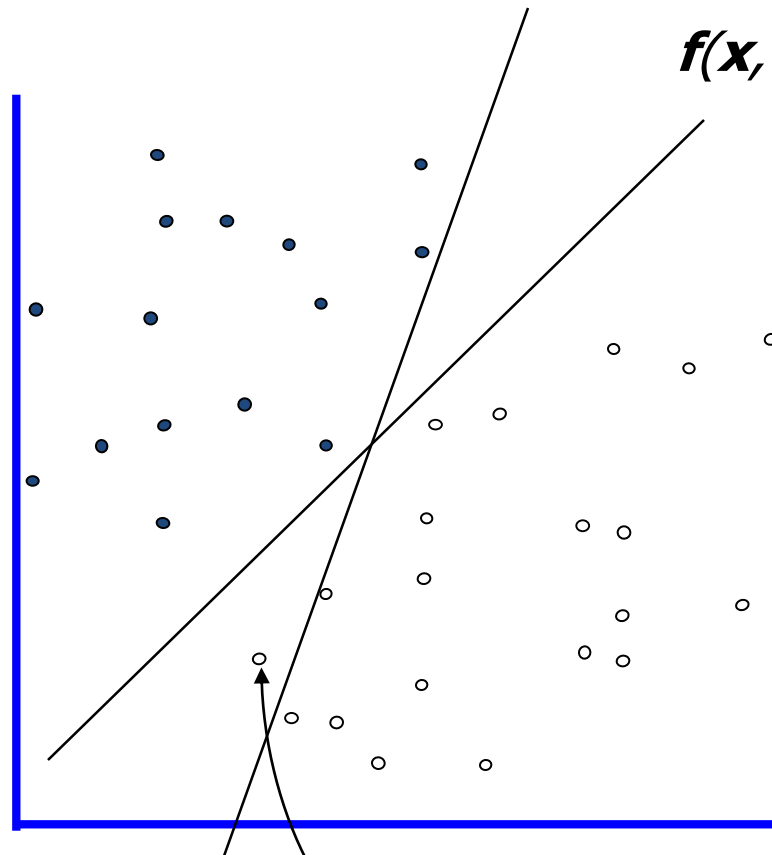
Any of these
would be fine..

..but which is
best?

- We may note that so far the classification error is concerned (with training data), all of them are with zero error.
- However, there is no guarantee that all hyperplanes perform equally well on unseen (i.e., test) data.

Linear Classifiers

- denotes +1
- denotes -1



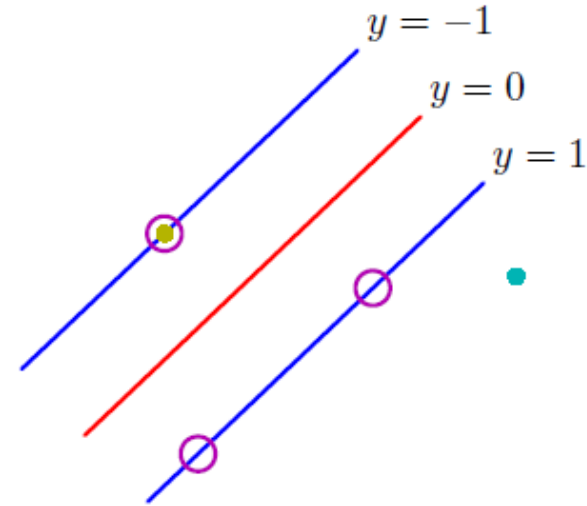
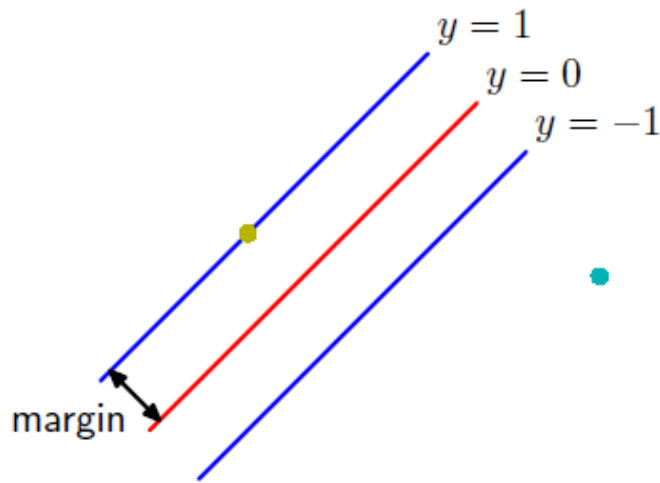
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \mathbf{x} + b)$$

How would you classify this data?

Misclassified
to +1 class

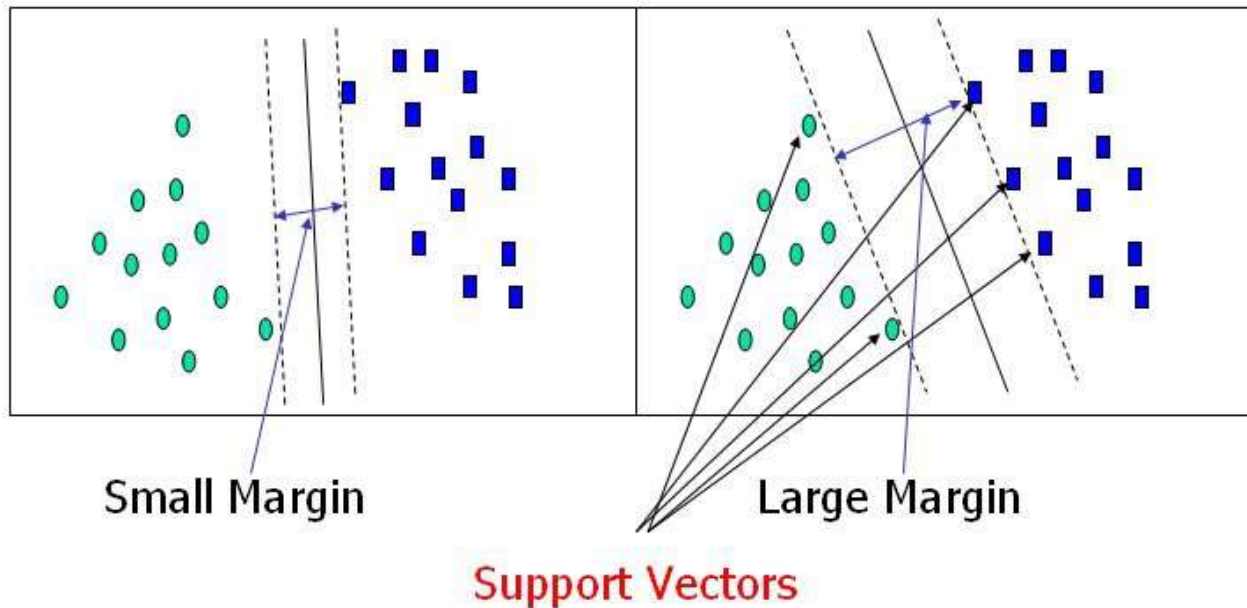
Linear SVM

Decision Boundary - SVM



- The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points, as shown on the left figure.
- Maximizing the margin leads to a particular choice of decision boundary, as shown on the right.
- The location of this boundary is determined by a subset of the data points, known as **support vectors**, which are indicated by the circles.

Large margin and support vectors



1. classifier that contains hyperplane with a small margin are more susceptible to model over fitting and tend to classify with weak confidence on unseen data.
2. Thus during the training or learning phase, the approach would be to search for the hyperplane with maximum margin. Such a hyperplane is called maximum margin hyperplane

Support Vector Machines

- Want line that maximizes the margin.

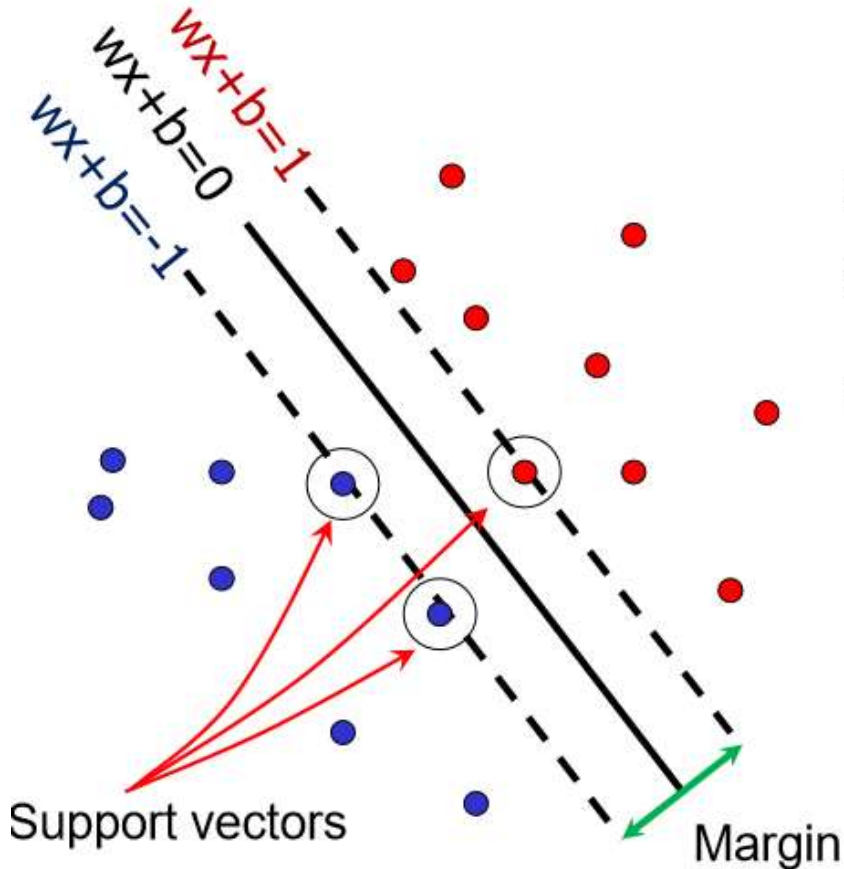
suppose that all the training data satisfy the following constraints:

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

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

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

- Find the \mathbf{W} (that defines the decision boundary) that correctly classifies the data with biggest possible margin**



- Essentially, the support vectors are the most difficult tuples to classify and give the most information regarding classification.**

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
- For the linearly separable case, the support vector algorithm simply looks for the separating hyperplane with largest margin.
- **Margin is a distance of a point(SV) and a plane**

Margin



Define the hyperplanes H such that:

$$w \cdot x_i + b \geq +1 \text{ when } y_i = +1$$

$$w \cdot x_i + b \leq -1 \text{ when } y_i = -1$$

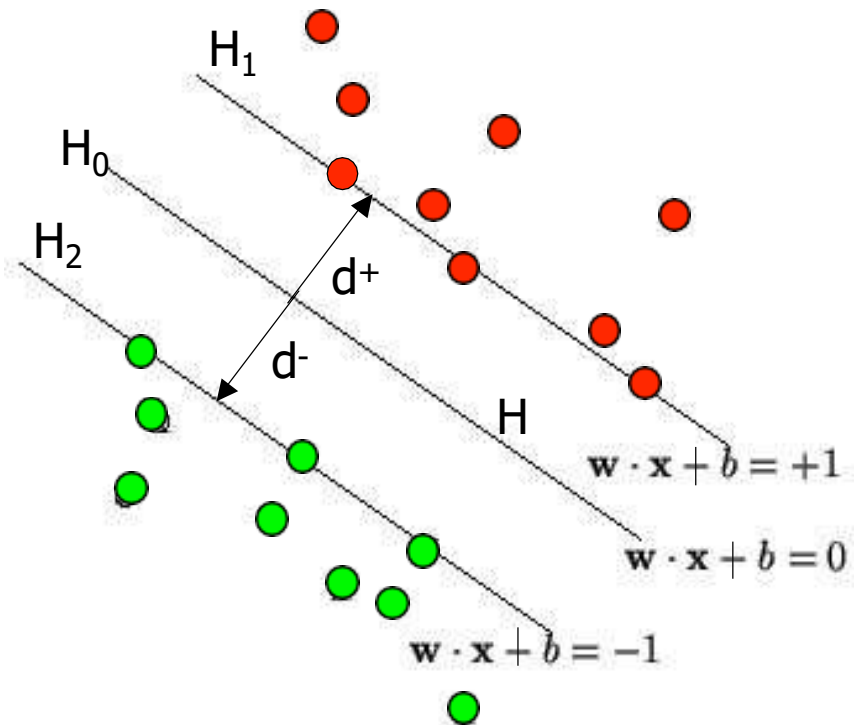
H_1 and H_2 are the planes:

$$H_1: w \cdot x_i + b = +1$$

$$H_2: w \cdot x_i + b = -1$$

The points on the planes H_1 and H_2 are the tips of the Support Vectors

The plane H_0 is the median in between, where $w \cdot x_i + b = 0$



$d+$ = the shortest distance to the closest positive point

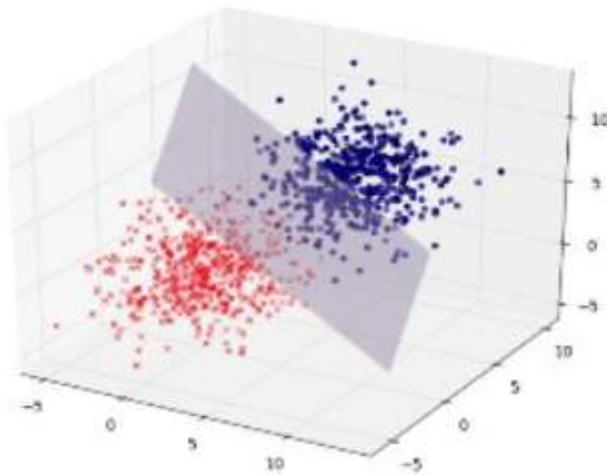
$d-$ = the shortest distance to the closest negative point

The margin (gutter) of a separating hyperplane is $d+ + d-$.

Example

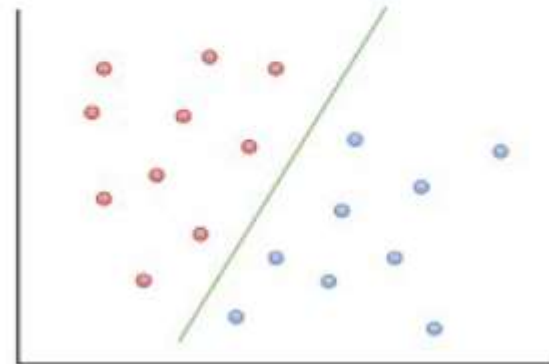
$$\mathbf{w}^T \mathbf{x} = 0$$

Hyperplane



$$y = ax + b$$

Line



Weight vector is perpendicular to the hyperplane

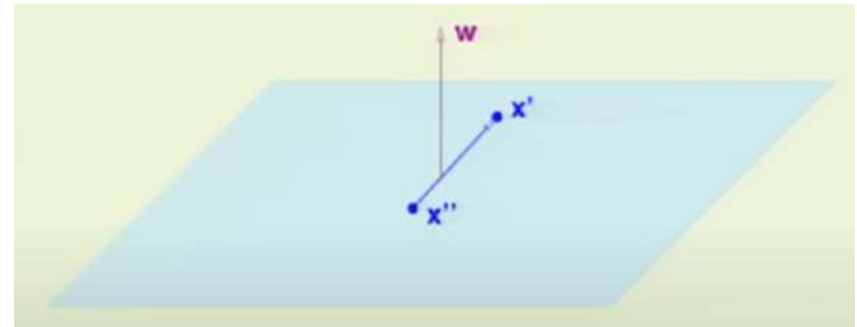


- Consider points x_a and x_b , which lie on the decision boundary.

- This gives us two equations:

$$w^T x_a + b = 0$$

$$w^T x_b + b = 0$$



- Subtracting these two equations gives us

$$w^T \cdot (x_a - x_b) = 0$$

Note that the vector $x_a - x_b$ lies on the decision boundary, and it is directed from x_b to x_a .

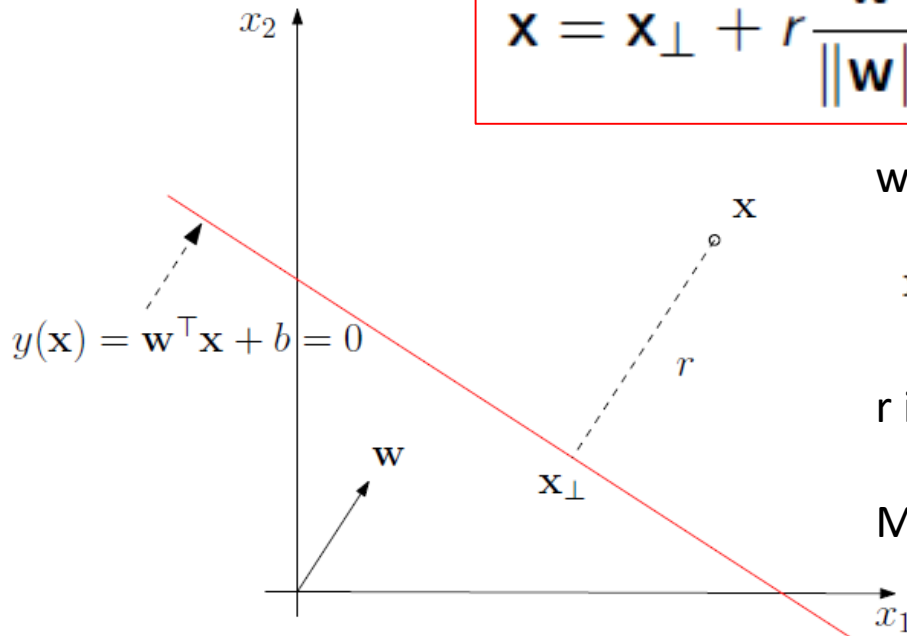
- Since the dot product $w^T \cdot (x_a - x_b)$ is zero, **w^T must be orthogonal to $x_a - x_b$ and in turn, to the decision boundary.**

Distance of a point to a line - orthogonal projection



$$\mathbf{x} = \mathbf{x}_{\perp} + r \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

← by vector addition



where $\frac{\mathbf{w}}{\|\mathbf{w}\|}$ = unit vector in the direction of \mathbf{w}

\mathbf{x}_{\perp} = orthogonal projection of \mathbf{x} onto line $y(\mathbf{x}) = 0$

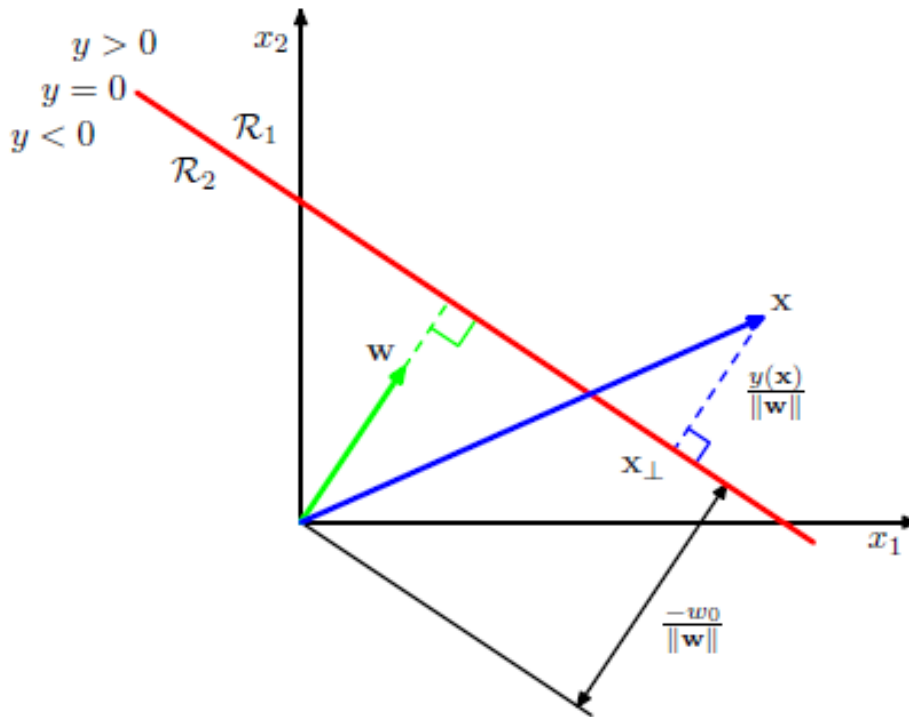
r is the (signed) distance of \mathbf{x} to the line.

Multiply left and right by \mathbf{w}^T and add b :

$$\underbrace{\mathbf{w}^T \mathbf{x} + b}_{y(\mathbf{x})} = \underbrace{\mathbf{w}^T \mathbf{x}_{\perp} + b}_0 + r \frac{\mathbf{w}^T \mathbf{w}}{\|\mathbf{w}\|}$$

$$r = y(\mathbf{x}) \frac{\|\mathbf{w}\|}{\|\mathbf{w}\|^2} = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$$

Distance to decision boundary



- The decision surface, shown in red, is perpendicular to \mathbf{w}
- its displacement from the origin is controlled by the bias parameter w_0 .
- if \mathbf{x} is a point on the decision surface, then $y(\mathbf{x}) = 0$, and so the normal distance from the origin to the decision surface is given by

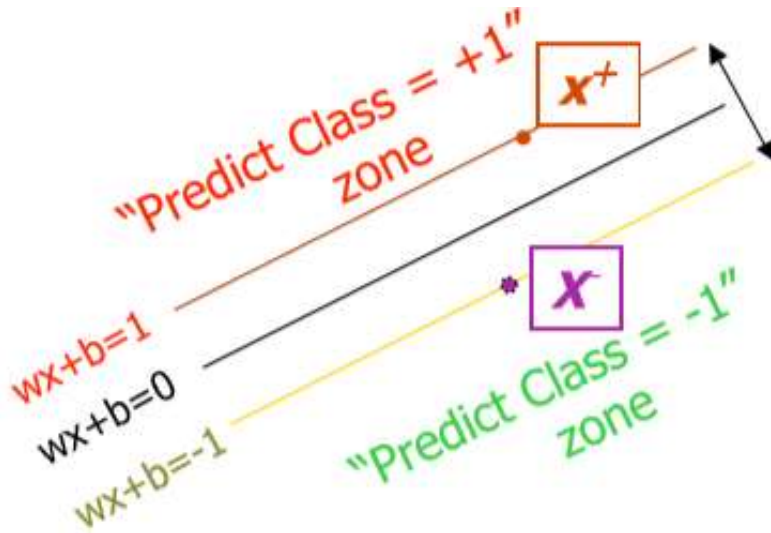
$$\frac{\mathbf{w}^T \mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}.$$

- Also, the signed orthogonal distance of a general point \mathbf{x} from the decision surface is given by

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}.$$

$$\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w} = w_1^2 + \dots + w_{M-1}^2$$

Calculating Margin



M=Margin Width

$$w \cdot x^+ + b = +1$$

$$w \cdot x^- + b = -1$$

Margin width = $D(X^+) + D(X^-)$

$$= \frac{|w \cdot x^+ + b|}{||w||} + \frac{|w \cdot x^- + b|}{||w||}$$

$$= \frac{|1|}{||w||} + \frac{|-1|}{||w||} = \frac{2}{||w||}$$

Distance between lines given by solving linear equation:

Maximize margin: $M = \frac{2}{||w||}$ \rightarrow **Equivalent to minimize: $\frac{1}{2} ||w||^2$**

Constraints

- In order to maximize the margin, we thus need to minimize : $\frac{1}{2} ||\mathbf{w}||^2$
- With the condition that there are no datapoints between H_1 and H_2 :
 - $\mathbf{x}_i \bullet \mathbf{w} + b \geq +1$ when $y_i = +1$
 - $\mathbf{x}_i \bullet \mathbf{w} + b \leq -1$ when $y_i = -1$
 - **Can be combined into: $y_i(\mathbf{x}_i \bullet \mathbf{w}) \geq 1$**

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

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

$$\text{same as } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

Solving the Optimization Problem



Find \mathbf{w} and b such that

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

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:

Optimization using Langrangian

Optimization Problem

- Optimization problem is typically written:

Minimize $f(x)$

subject to

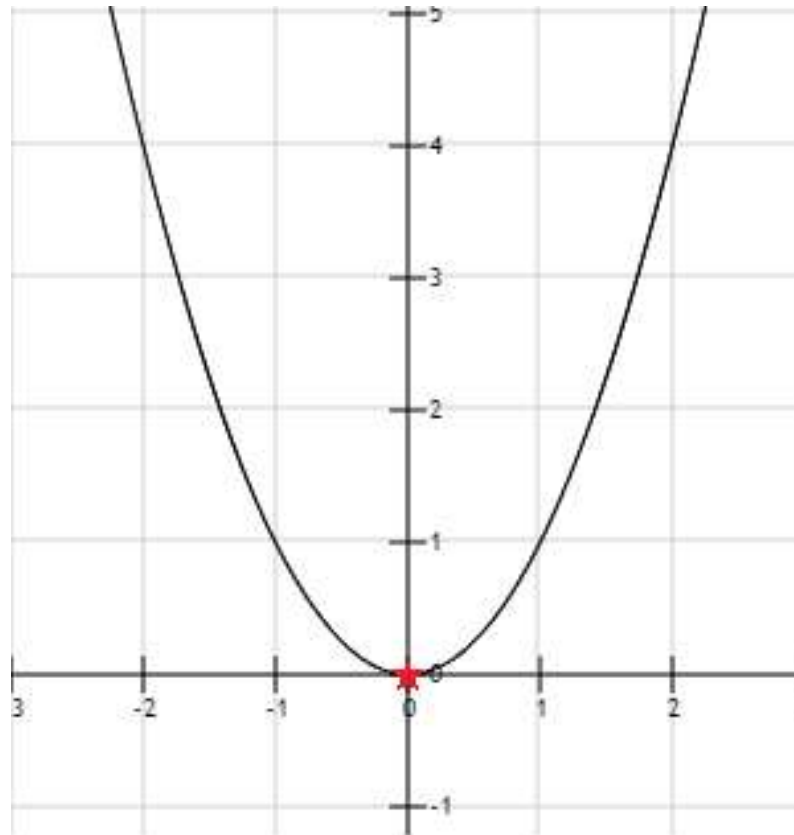
$$g_i(x) = 0, \quad i=1,\dots,p$$

$$h_i(x) \leq 0, \quad i=1,\dots,m$$

- $f(x)$ is called the objective function
- By changing x (the optimization variable) we wish to find a value x^* for which $f(x)$ is at its minimum.
- p functions of g_i define equality constraints and
- m functions h_i define inequality constraints.
- The value we find **MUST** respect these constraints!

Unconstrained Optimization

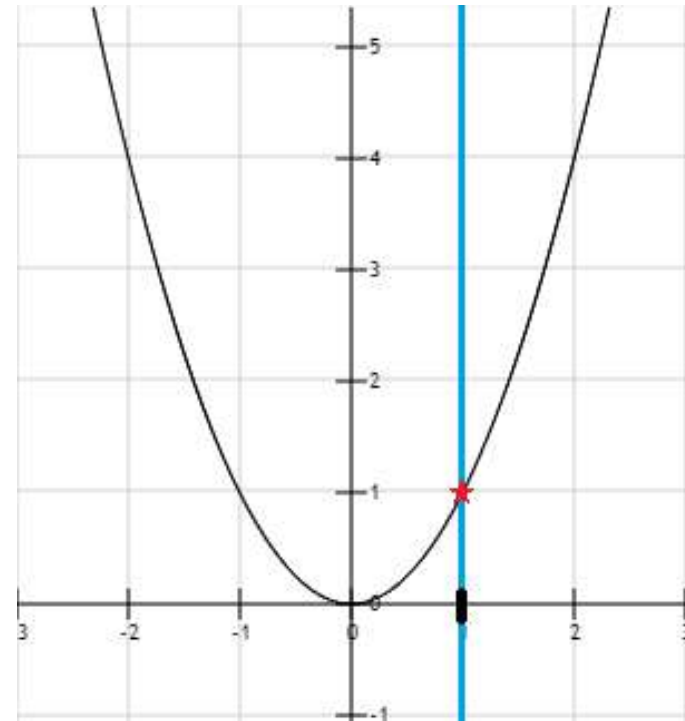
- Minimize x^2



Constrained Optimization -Equality Constraint

Minimize x^2

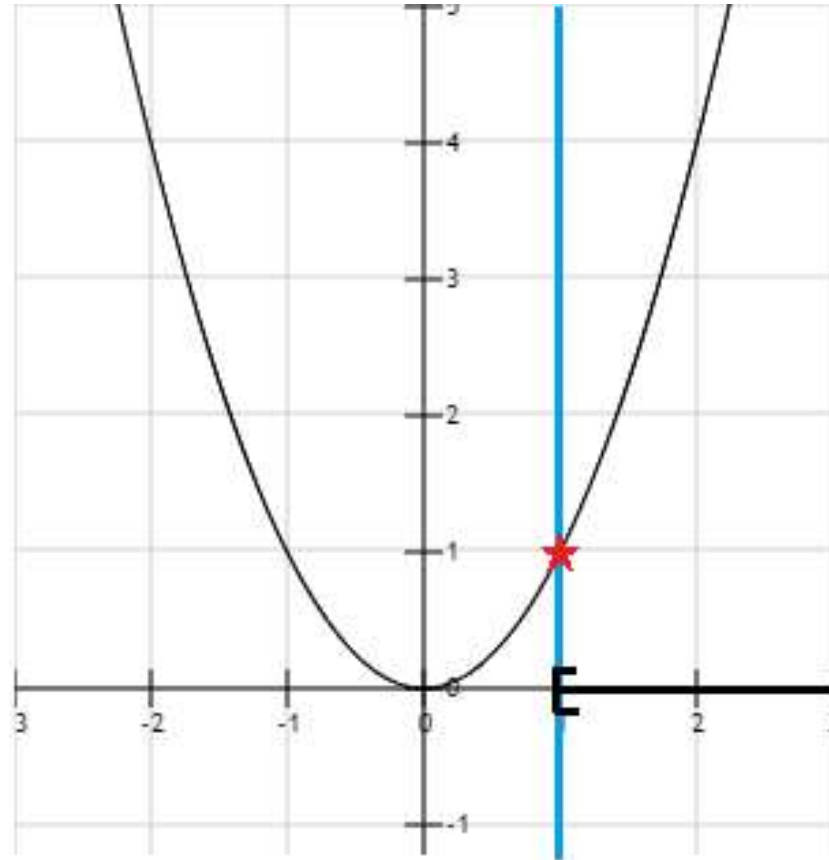
Subject to $x = 1$



Constrained Optimization -Inequality Constraint

Minimize x^2

Subject to $x \geq 1$



Constrained optimization

- We can also have mix equality and inequality constraints together.
- Only restriction is that if we use contradictory constraints, we can end up with a problem which does not have a feasible set

Minimize x^2

Subject to

$$x = 1$$

$$x < 0$$

Impossible for x to be equal 1 and less than zero at the same

Lagrange Multipliers

- **How do we find the solution to an optimization problem with constraints?**
- Constrained maximization (minimization) problem is rewritten as a Lagrange function
Lagrange function use Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to constraints

Example



Maximize

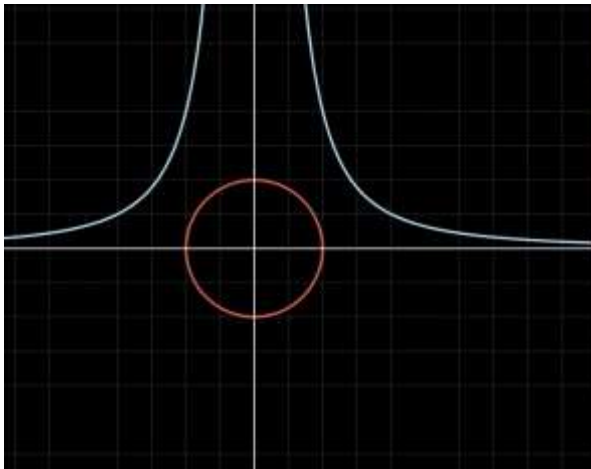
$$f(x,y) = x^2 y$$

Subject to

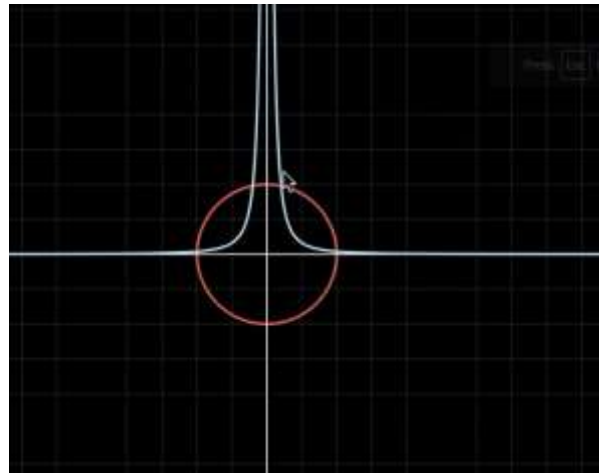
$$g(x, y) : x^2 + y^2 = 1$$

$$f(x,y) = K$$

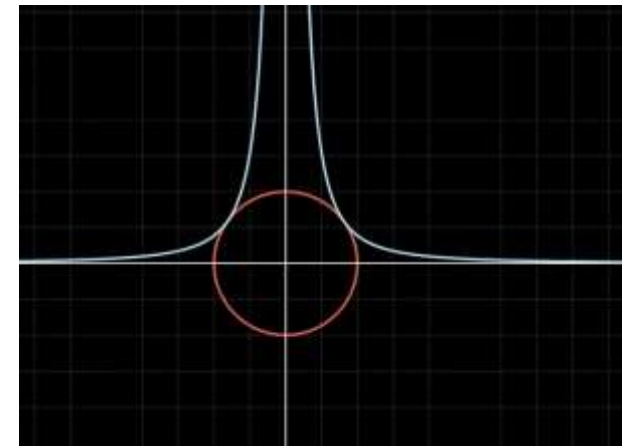
Larger constant K

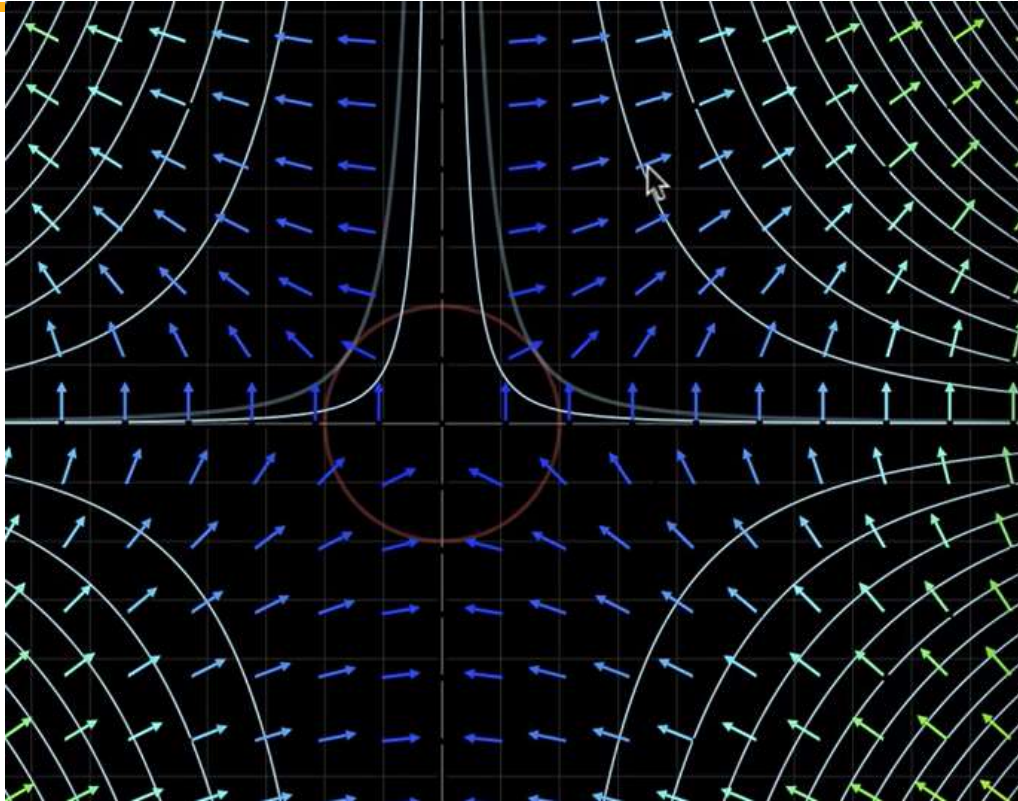


Smaller constant K



**Maximize the function for
some value of K**





- Gradients are perpendicular to various contour lines(w.r.t to different values of function) of function f
- Function is not changing the value along the points on any the contour line



Constrained to Unconstrained Optimization: Lagrange Multiplier

- Maximum of $f(x,y)$ under constraint $g(x,y)$ is obtained when their gradients point to same direction (when they are tangent to each other).

- Introduce a Lagrange multiplier λ for the equality constraint

- Mathematically,

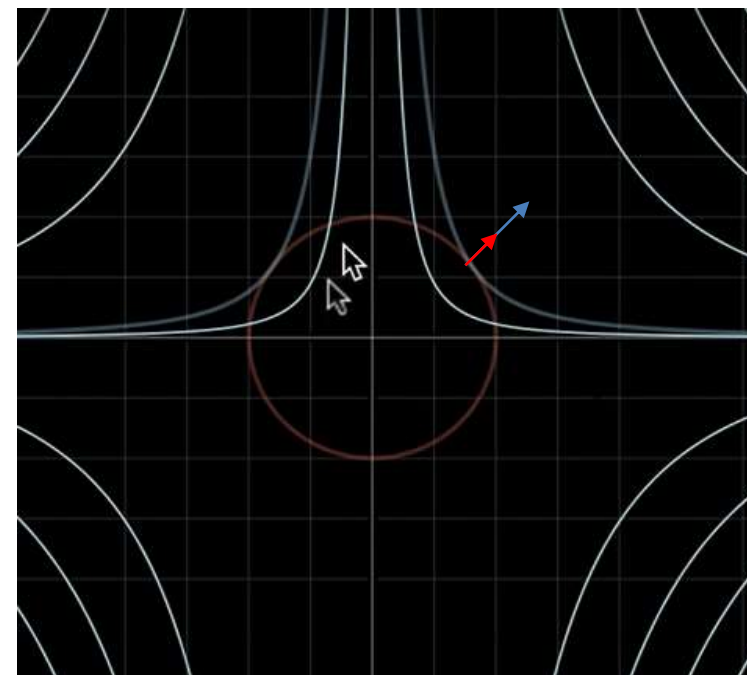
$$\nabla f(x,y) = \lambda \nabla g(x,y)$$

Equivalently:

$$L(x,y, \lambda) = f(x,y) - \lambda \cdot g(x,y) - C$$

Optimize L using gradients

$$\nabla L(x,y, \lambda) = 0$$



Summary - Lagrange multiplier method



1. Construct the Lagrangian function by introducing one multiplier per constraint
2. Get the gradient ∇L of the of the Lagrangian
3. Solve for $\nabla L(x,y, \lambda)=0$

Example:

$$\max_{x,y} xy \text{ subject to } x + y = 6$$

- Introduce a Lagrange multiplier λ for constraint
- Construct the Lagrangian

$$L(x, y) = xy - \lambda(x + y - 6)$$

- Stationary points

$$\frac{\partial L(x, y)}{\partial \lambda} = x + y - 6 = 0$$

$$\left. \begin{aligned} \frac{\partial L(x, y)}{\partial x} &= y - \lambda = 0 \\ \frac{\partial L(x, y)}{\partial y} &= x - \lambda = 0 \end{aligned} \right\} \Rightarrow x = y = \lambda$$

$$\Rightarrow x = y = 3$$

x and y values remain same even if you take $+\lambda$ or $-\lambda$ for equality constraint

$$\begin{aligned} 2x &= 6 \\ x &= y = 3 \\ \lambda &= 3 \end{aligned}$$



Karush–Kuhn–Tucker (KKT) theorem

- KKT approach to nonlinear programming (quadratic) generalizes the method of [Lagrange multipliers](#), which allows only equality constraints.
- KKT allows inequality constraints



Karush-Kuhn-Tucker (KKT) conditions

- Start with

max $f(x)$ subject to

$$g_i(x) = 0 \text{ and } h_j(x) \geq 0 \text{ for all } i, j$$

- f is the objective function, g_i ($i=1 \dots m$) are the equality constraint functions and h_j ($j=1 \dots l$) are the inequality constraint functions

- Make the Lagrangian function

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

- Take gradient and set to 0 – but satisfy other conditions also.

KKT conditions

- 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

Stationarity

$$\nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) = 0$$

primal feasibility condition

$$g_i(x^*) = 0 \text{ for all } i$$

primal feasibility condition

$$h_j(x^*) \geq 0 \text{ for all } j$$

dual feasibility condition

$$\mu_j \geq 0 \text{ for all } j$$

complementary slackness condition

$$\mu_j^* h_j(x^*) = 0 \text{ for all } j$$

Solving SVM using KKT conditions

SVM objective function

max $f(x)$ subject to

$$g_i(x) = 0 \text{ and } h_j(x) \geq 0 \text{ for all } i, j$$

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

Find \mathbf{w} and b such that

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

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

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



Solving the Optimization Problem

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

KKT condition : Stationarity

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

Solving the Optimization Problem

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

- Expanding above equation:

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

- Substituting $W = \sum \alpha_i y_i x_i$ and $\sum \alpha_i y_i = 0$ in above equation

$$L(w, b, \alpha_i) = \frac{1}{2} (\sum_i \alpha_i y_i x_i) (\sum_j \alpha_j y_j x_j) - (\sum_i \alpha_i y_i x_i) (\sum_j \alpha_j y_j x_j) + \sum \alpha_i$$

$$L(w, b, \alpha_i) = \sum \alpha_i - \frac{1}{2} (\sum_i \alpha_i y_i x_i) (\sum_j \alpha_j y_j x_j)$$

$$L(w, b, \alpha_i) = \sum \alpha_i - \frac{1}{2} (\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i \cdot x_j) \rightarrow \text{Wolfe dual Lagrangian function.}$$



The Dual Problem

- This is constrained optimization problem , popularly known as convex optimization problem, where objective function is quadratic and constraints are linear in the parameters W and b . The original problem is known as the **primal problem**
- **The solution involves constructing a *dual problem* where a *Lagrange multiplier* is associated with every constraint in the primary problem:** The new objective function is in terms of α_i only
- It is known as the dual problem: if we know \mathbf{w} , we know all α_i ; if we know all α_i , we know \mathbf{w} . The objective function of the dual problem needs to be maximized (comes out from the KKT theory)
- The **duality principle** tells us that an optimization problem can be viewed from two perspectives. The first one is the primal problem, a minimization problem in our case, and the other one is the dual problem, which will be a maximization problem.

The Dual Problem

$$\underset{\alpha}{\text{maximize}} \quad \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

subject to $\alpha_i \geq 0$, for any $i = 1, \dots, m$ → Properties of α_i when we introduce the Lagrange multipliers

$$\sum_{i=1}^m \alpha_i y_i = 0$$

→ The result when we differentiate the original Lagrangian w.r.t. b

When we solve the Wolfe dual problem, we get a vector α containing all Lagrange multipliers.



Lagrange multiplier for support vectors

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

- Using KKT **complementary slackness** condition
 $\alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1] = 0$

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 $\alpha_i > 0$

- Using dual feasibility condition** $\alpha_i \geq 0$

case 1 : $\alpha_i > 0$

- For support vectors: $\{\mathbf{w}^T \mathbf{x}_i + b = +1 \text{ when } Y=1 \text{ or } : \mathbf{w}^T \mathbf{x}_i + b = -1 \text{ when } Y=-1\}$
 $y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0$

case 2 : $\alpha_i = 0$

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

Linear SVM summary – 1

SVM Optimization Problem – primal



Find \mathbf{w} and b such that

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

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

→ Objective function

$L(\mathbf{w}, b, \alpha_i) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum \alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1]$ → Lagrangian function

$$\min_{\mathbf{w}, b} \max_{\alpha} \mathcal{L}(\mathbf{w}, b, \alpha)$$

subject to $\alpha_i \geq 0, i = 1, \dots, m$

→ Lagrangian primal problem

Solving the minimization problem involves taking the partial derivatives of L with respect to \mathbf{W} and b .

Linear SVM summary – 2

SVM Optimization Problem - dual



$$\begin{aligned} &\underset{\alpha}{\text{maximize}} && \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ &\text{subject to} && \alpha_i \geq 0, \text{ for any } i = 1, \dots, m \\ &&& \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

→ Wolfe dual problem

- solve the Wolfe dual problem, we get a vector α containing all Lagrange multipliers.
- Compute w : Taking partial derivative with respect to w
- $\frac{\partial L}{\partial w} = \mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i$
- Compute b : For support vectors $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$

Linear SVM summary – 3

Hypothesis function



- The SVMs use the same hypothesis function as the Perceptron. The class of an example is given by:

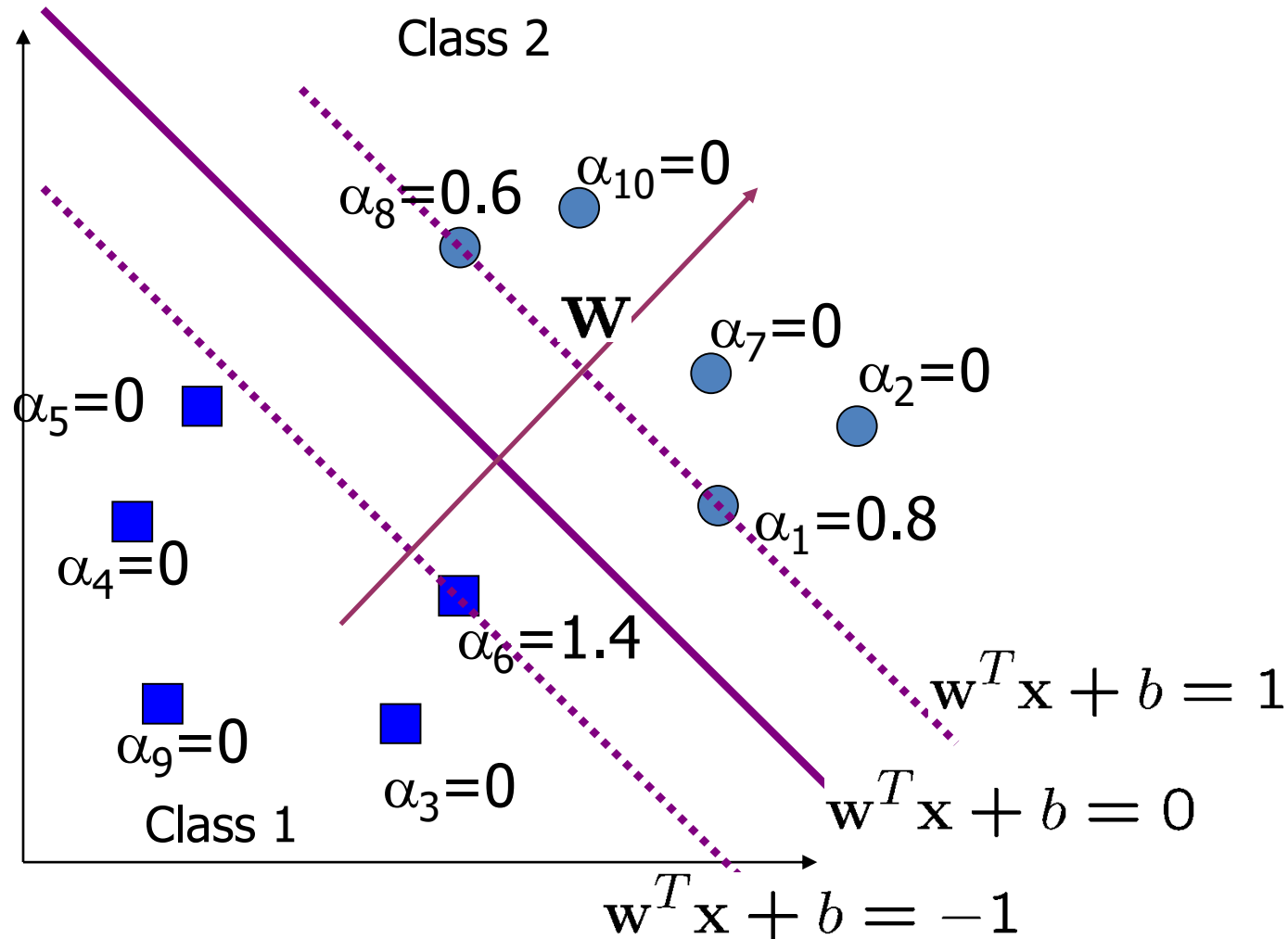
$$h(\mathbf{x}_i) = \text{sign}(\mathbf{w} \cdot \mathbf{x}_i + b)$$

- When using the dual formulation, it is computed using only the support vectors:

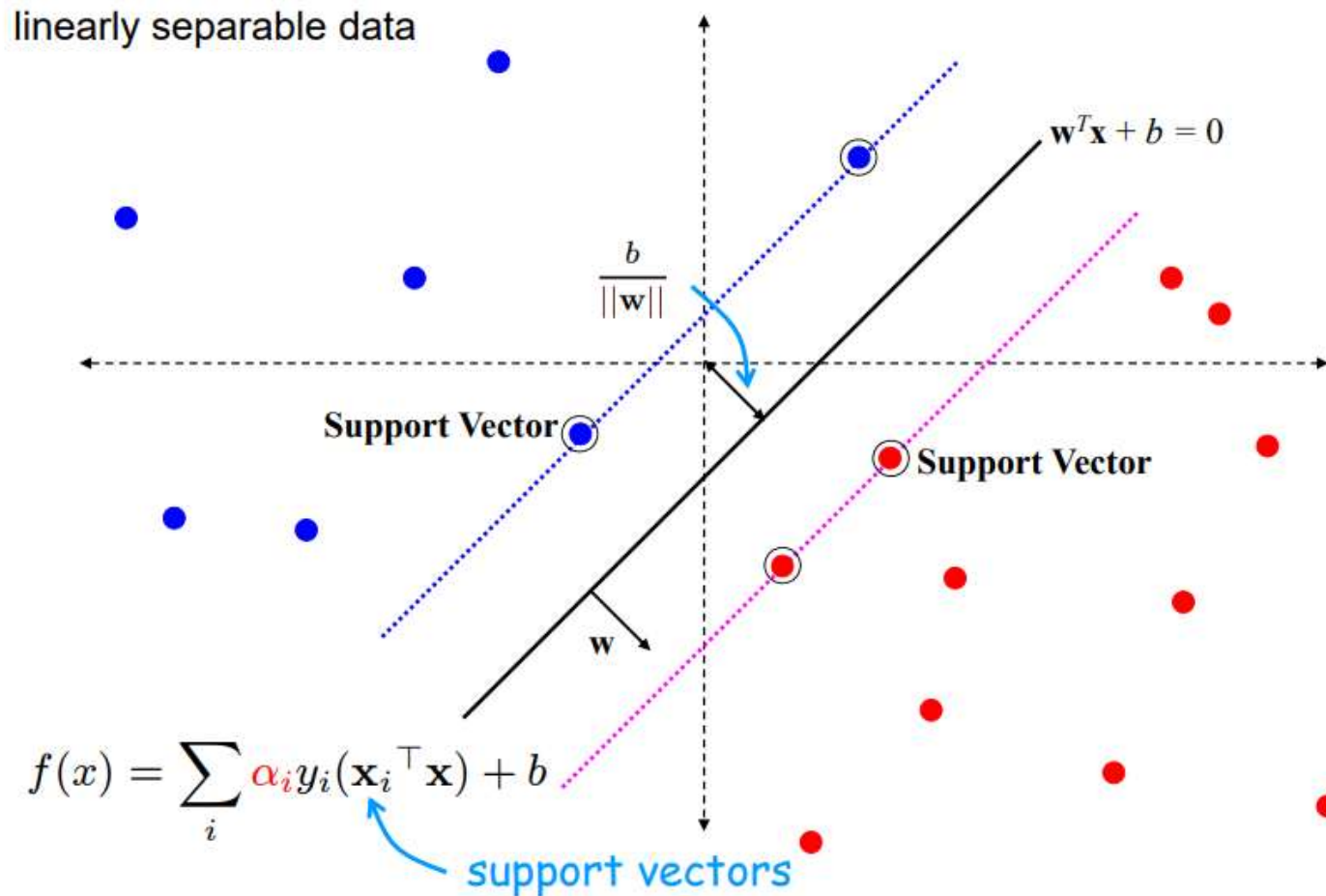
$$h(\mathbf{x}_i) = \text{sign}\left(\sum_{j=1}^S \alpha_j y_j (\mathbf{x}_j \cdot \mathbf{x}_i) + b\right)$$

- $\mathbf{x}_j \Rightarrow$ support vectors
- $\mathbf{x}_i \Rightarrow$ Test example

A Geometrical Interpretation



Substituting w in support vectors function




SVM

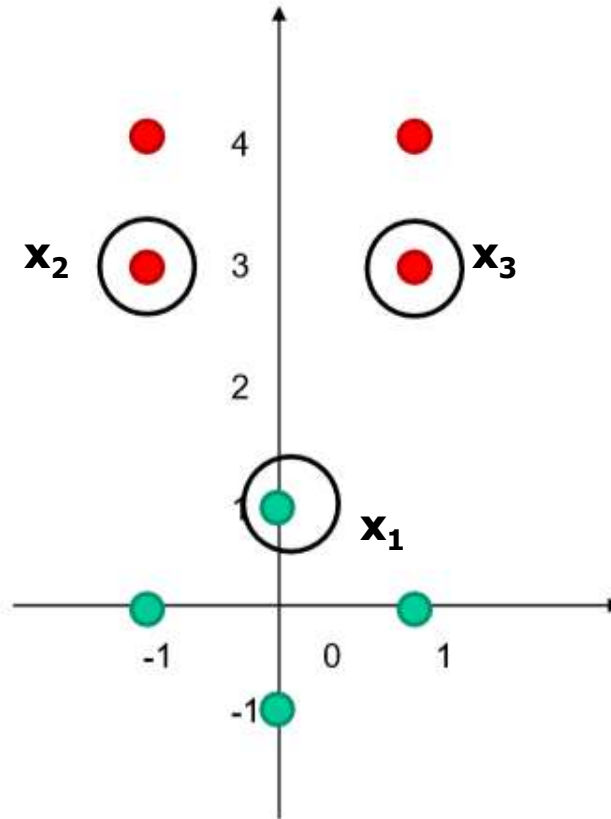


- Once the SVM is trained with training data, the complexity of the classifier is characterized by the number of support vectors.
- Dimensionality of data is not an issue in SVM unlike in other classifier.

Example



 = support vectors



POS

NEG

Example adapted from Dan Ventura

Solving for α



- We know that for the support vectors, $f(x) = 1$ or -1 exactly
- Add a 1 in the feature representation for the bias
- The support vectors have coordinates and labels:
 - $\mathbf{x1} = [0 \ 1 \ 1]$, $y1 = -1$
 - $\mathbf{x2} = [-1 \ 3 \ 1]$, $y2 = +1$
 - $\mathbf{x3} = [1 \ 3 \ 1]$, $y3 = +1$
- Thus we can form the following system of linear equations:

Solving for α



$$f(x) = \sum_i \alpha_i y_i (\mathbf{x}_i^\top \mathbf{x}) + b$$

- System of linear equations:

$$\alpha_1 y_1 \text{dot}(\mathbf{x}_1, \mathbf{x}_1) + \alpha_2 y_2 \text{dot}(\mathbf{x}_1, \mathbf{x}_2) + \alpha_3 y_3 \text{dot}(\mathbf{x}_1, \mathbf{x}_3) = y_1$$

$$\alpha_1 y_1 \text{dot}(\mathbf{x}_2, \mathbf{x}_1) + \alpha_2 y_2 \text{dot}(\mathbf{x}_2, \mathbf{x}_2) + \alpha_3 y_3 \text{dot}(\mathbf{x}_2, \mathbf{x}_3) = y_2$$

$$\alpha_1 y_1 \text{dot}(\mathbf{x}_3, \mathbf{x}_1) + \alpha_2 y_2 \text{dot}(\mathbf{x}_3, \mathbf{x}_2) + \alpha_3 y_3 \text{dot}(\mathbf{x}_3, \mathbf{x}_3) = y_3$$

$$-2 * \alpha_1 + 4 * \alpha_2 + 4 * \alpha_3 = -1$$

$$-4 * \alpha_1 + 11 * \alpha_2 + 9 * \alpha_3 = +1$$

$$-4 * \alpha_1 + 9 * \alpha_2 + 11 * \alpha_3 = +1$$

$$\alpha_i [-1 (\mathbf{w} \cdot \mathbf{x}_i + b)] = -1$$

$$\alpha_i [+1 (\mathbf{w} \cdot \mathbf{x}_i + b)] = 1$$

- Solution: $\alpha_1 = 3.5$, $\alpha_2 = 0.75$, $\alpha_3 = 0.75$

Solving for w, b; plotting boundary

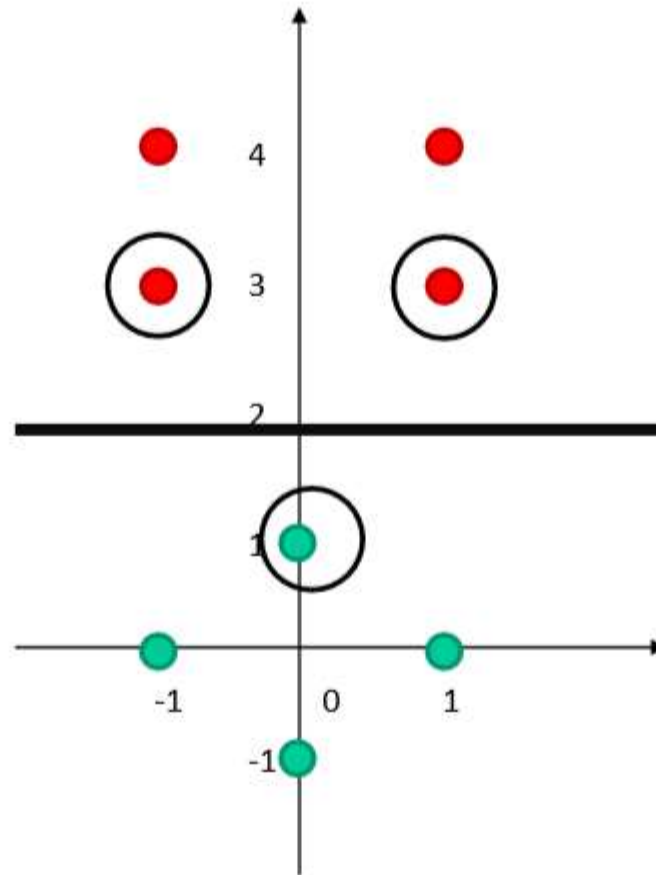


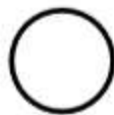
- We know $\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i$ i.e $\mathbf{w} = \alpha_1 y_1 \mathbf{x}_1 + \dots + \alpha_N y_N \mathbf{x}_N$
where N = No of SVs
- Thus $\mathbf{w} = -3.5 * [0 \ 1 \ 1] + 0.75 [-1 \ 3 \ 1] + 0.75 [1 \ 3 \ 1] = [0 \ 1 \ -2]$
- Separating out weights and bias, we have: $\mathbf{w} = [0 \ 1]$ and $b = -2$
 $a=0, c=1$

Boundary:

- For SVMs, we used this eq for a line: $ax + cy + b = 0$ where $\mathbf{w} = [a \ c]$
- Thus $ax + b = -cy \rightarrow y = (-a/c)x + (-b/c)$
- Thus y-intercept is $(-b/c) = -(-2)/1 = 2$
- The decision boundary is perpendicular to \mathbf{w} and it has slope
 $=(-a/c) = -0/1 = 0$

Decision boundary



 = support vectors

POS

DECISION BOUNDARY

NEG



Good Web References for SVM

- **Text categorization with Support Vector Machines: learning with many relevant features** - T. Joachims, ECML
- **A Tutorial on Support Vector Machines for Pattern Recognition**, Kluwer Academic Publishers - Christopher J.C. Burges
- <http://www.cs.utexas.edu/users/mooney/cs391L/>
- <https://www.coursera.org/learn/machine-learning/home/week/7>
- <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>
- <https://data-flair.training/blogs/svm-kernel-functions/>
- [MIT 6.034 Artificial Intelligence, Fall 2010](#)
- <https://stats.stackexchange.com/questions/30042/neural-networks-vs-support-vector-machines-are-the-second-definitely-superior>
- <https://www.sciencedirect.com/science/article/abs/pii/S0893608006002796>
- <https://medium.com/deep-math-machine-learning-ai/chapter-3-support-vector-machine-with-math-47d6193c82be>
- <https://www.svm-tutorial.com/>

Thank You