Support Vector Machines

Machine Learning Series

Jerry Jeychandra

Blohm Lab

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

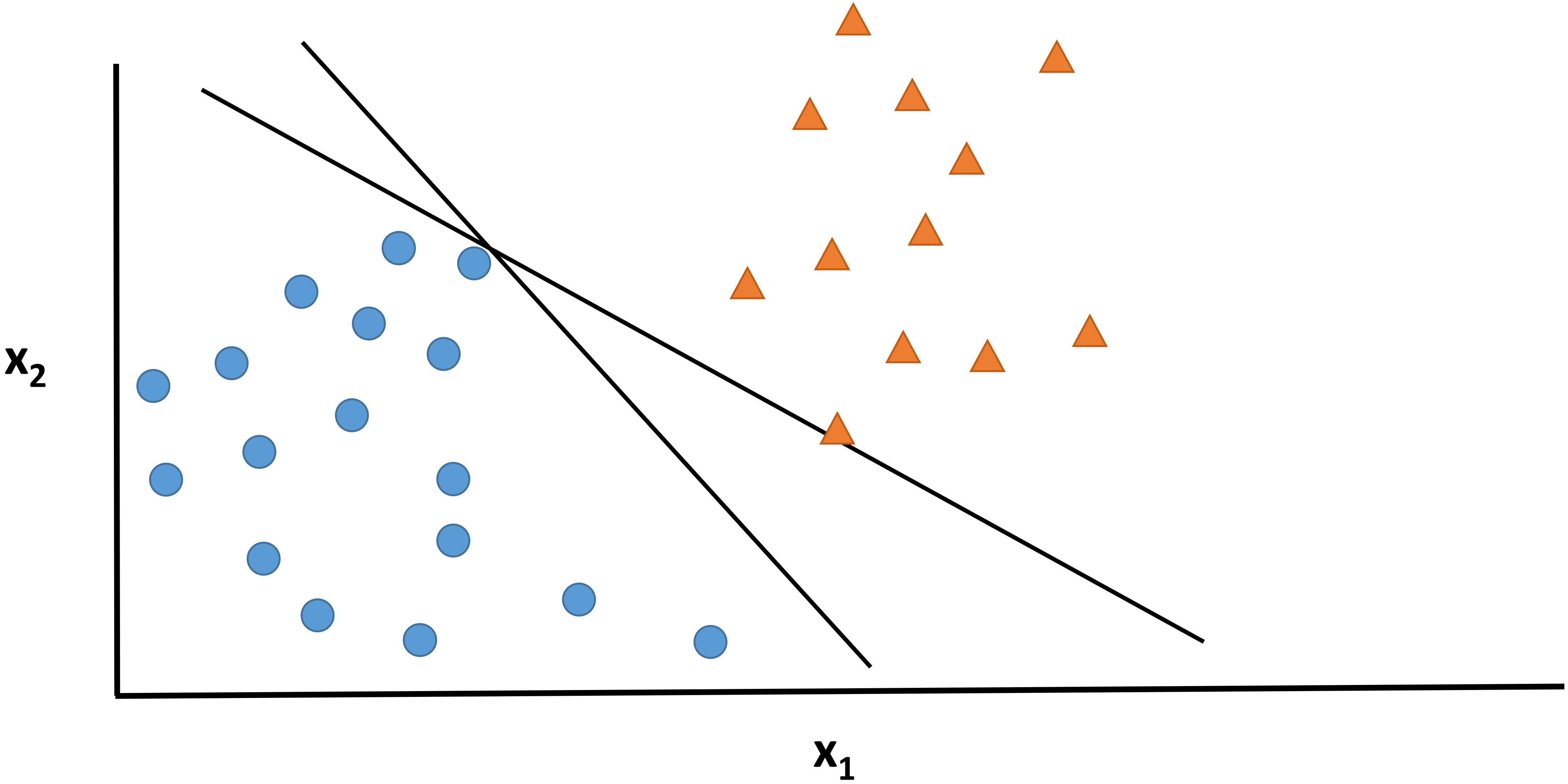
1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic Gradient Descent (extra)

Outline

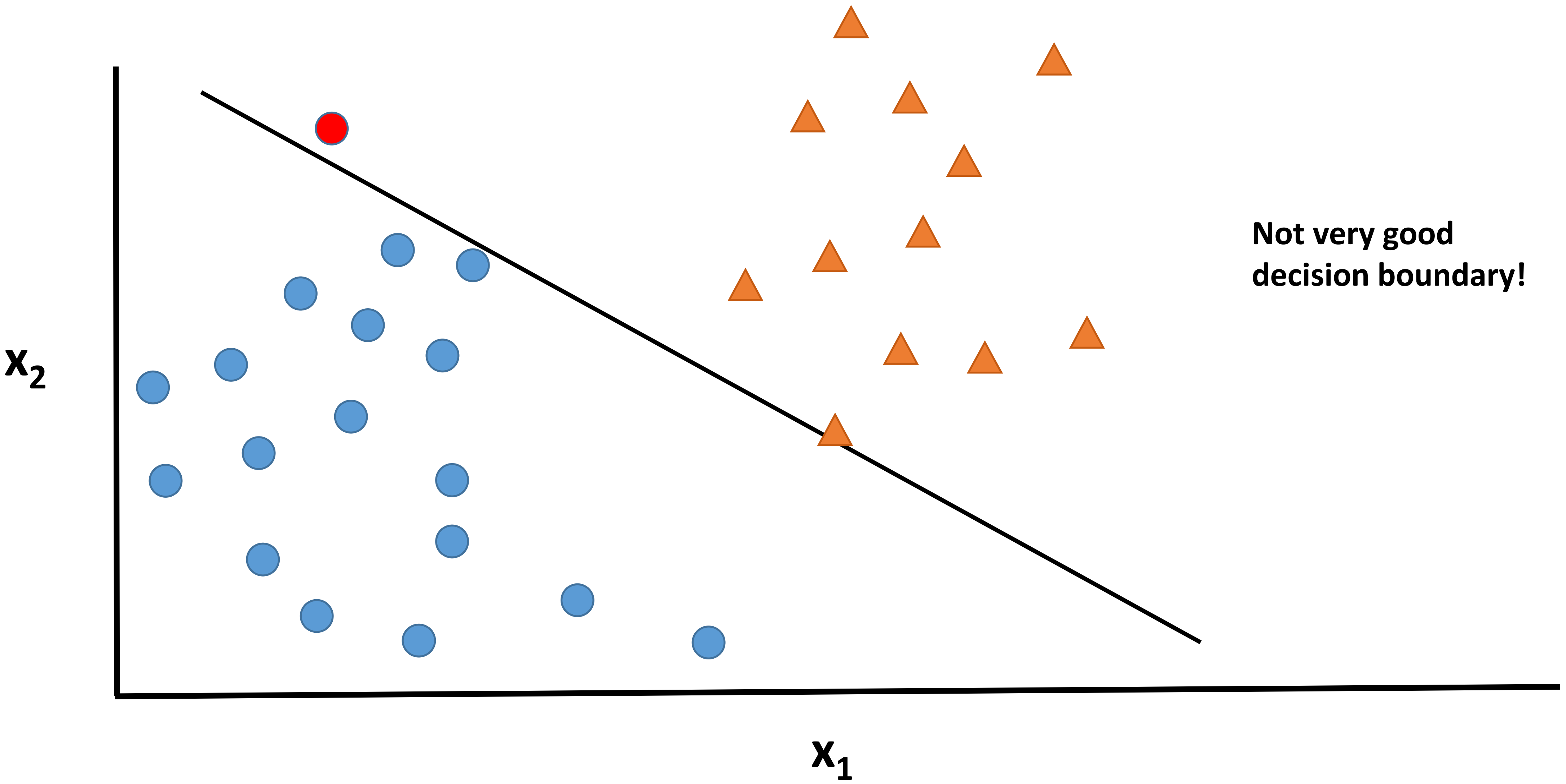
- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic Gradient Descent (extra)

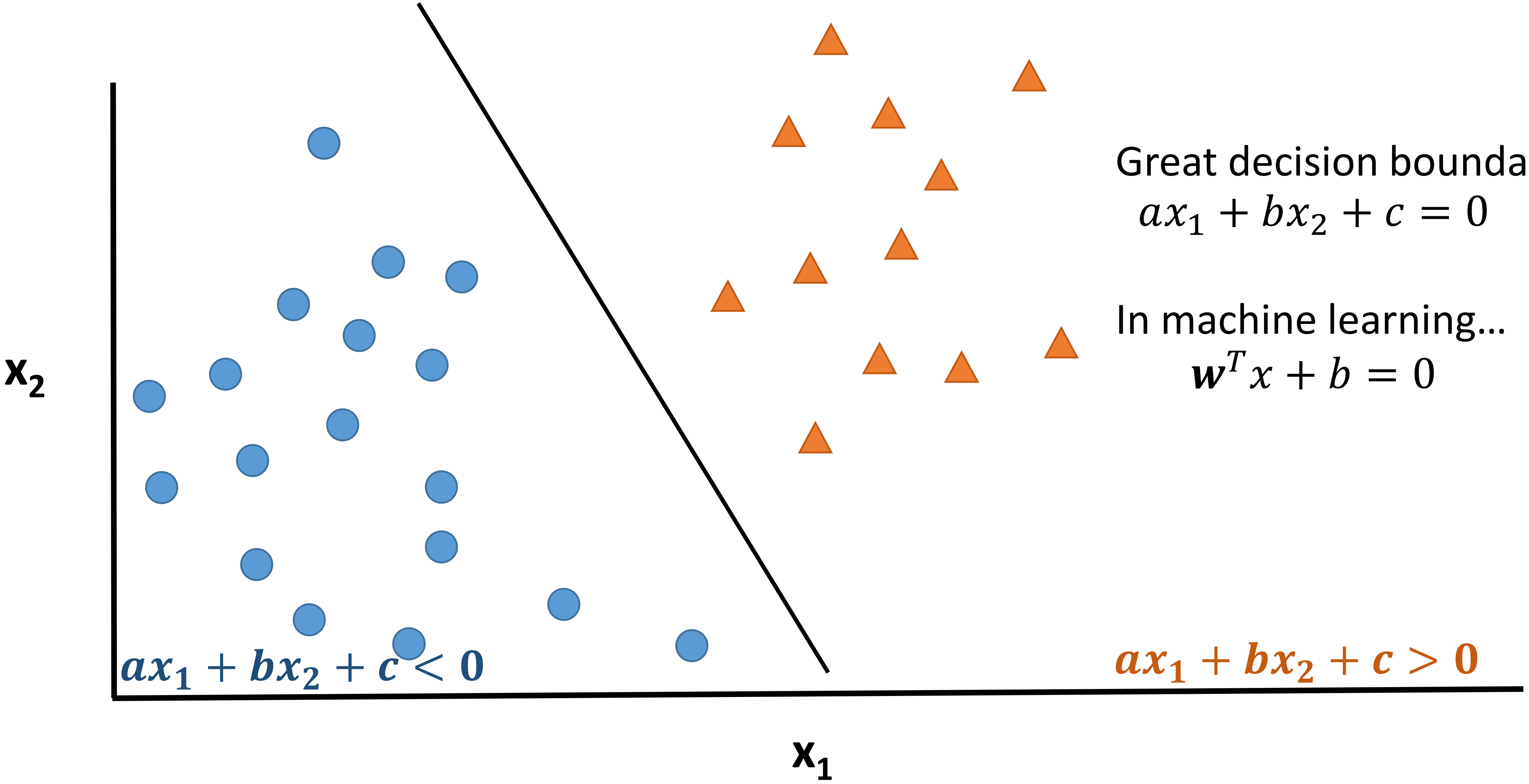
Support Vector Machines (SVM)



Support Vector Machines (SVM)



Support Vector Machines (SVM)



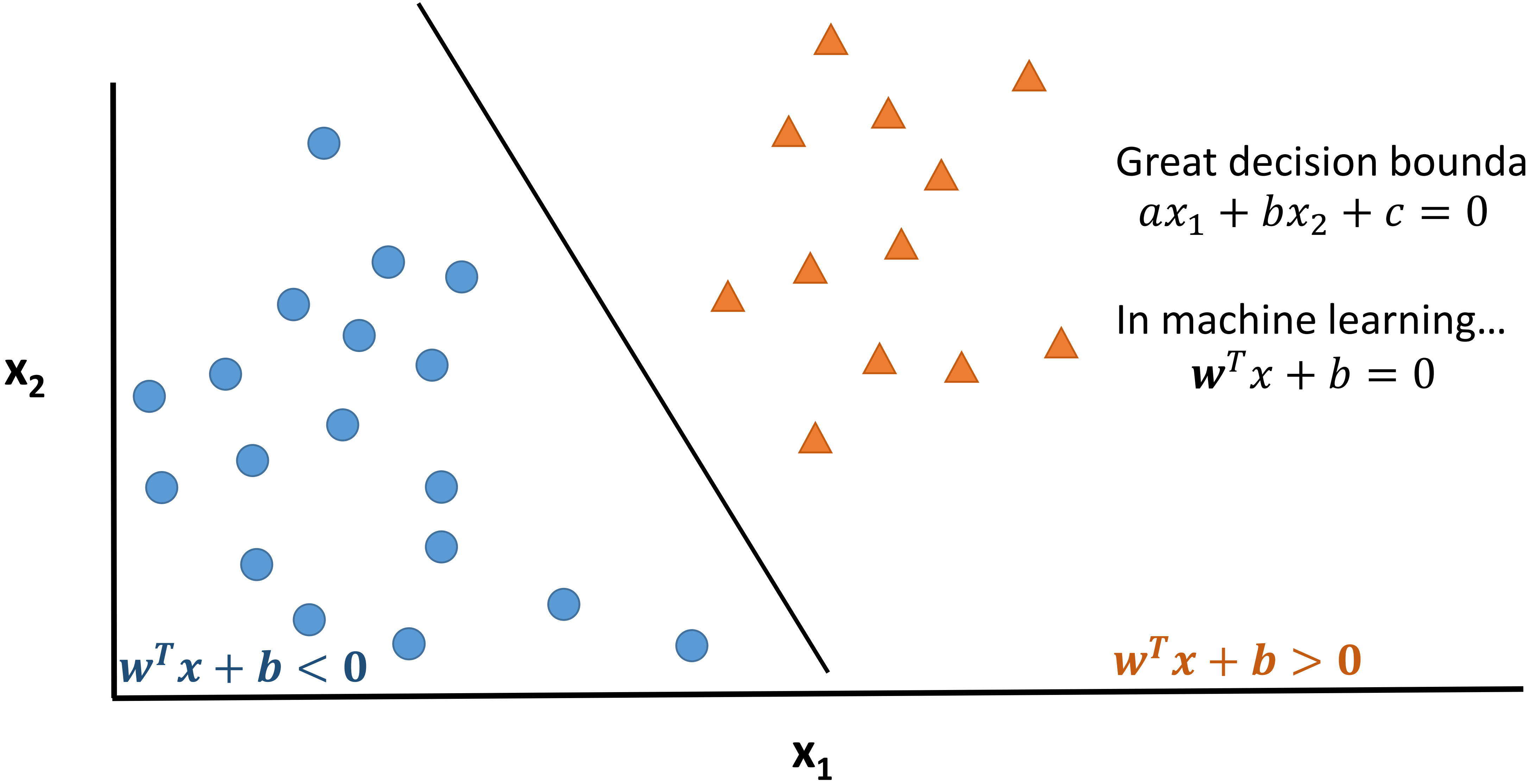
Great decision boundary!

$$ax_1 + bx_2 + c = 0$$

In machine learning…

$$\boldsymbol{w}^T x + b = 0$$

Support Vector Machines (SVM)



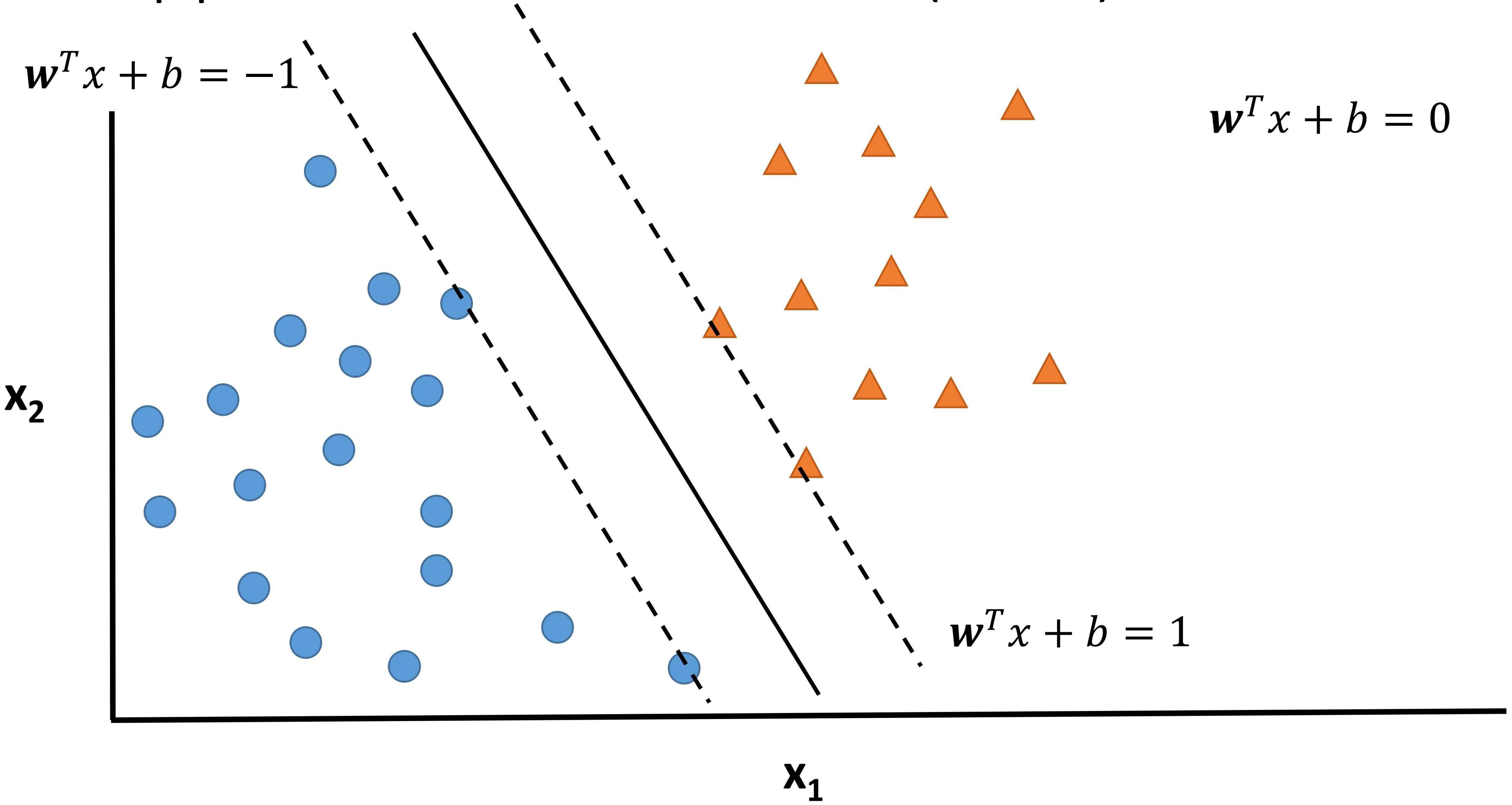
Great decision boundary!

$$ax_1 + bx_2 + c = 0$$

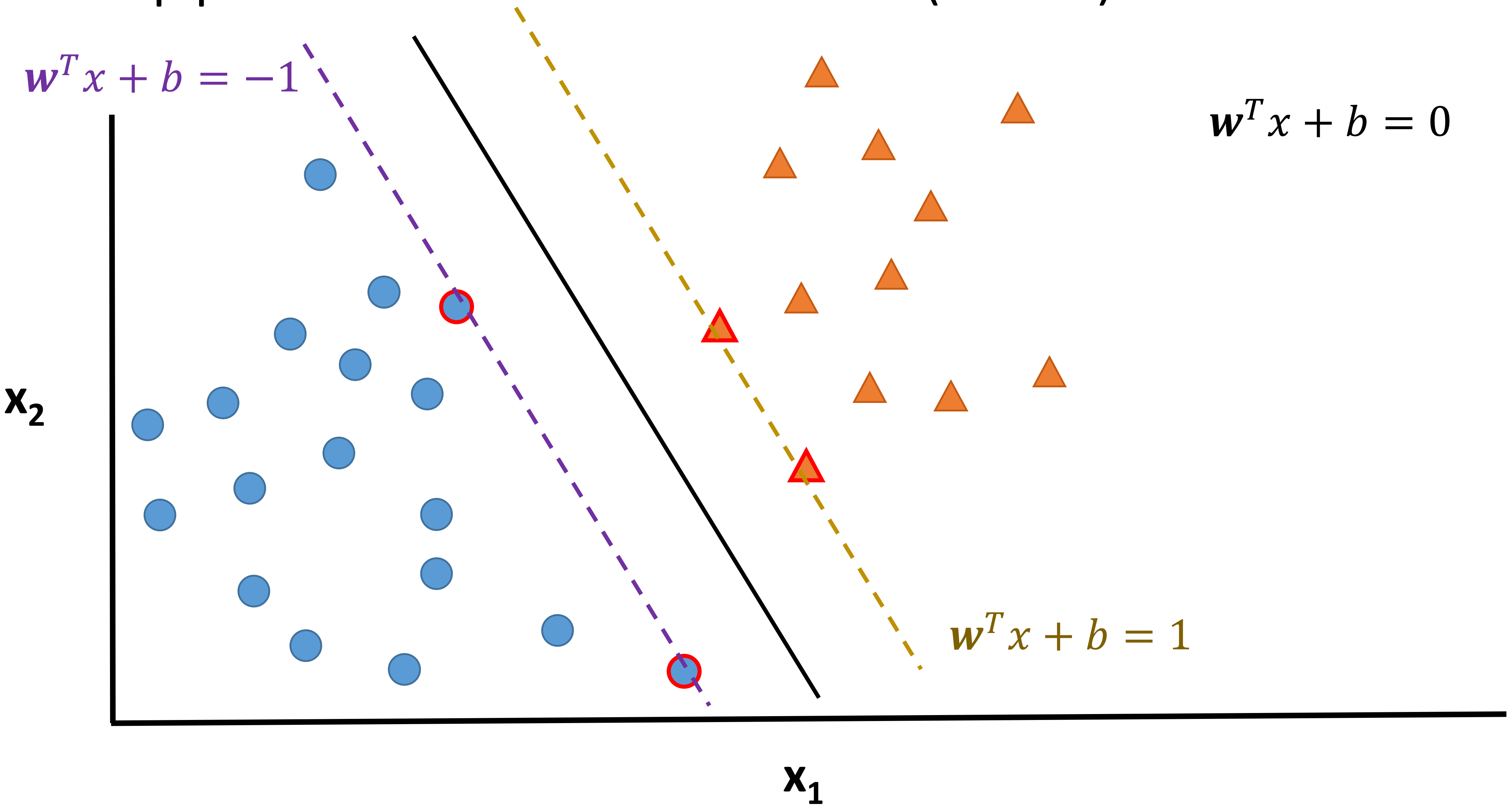
In machine learning…

$$\boldsymbol{w}^T x + b = 0$$

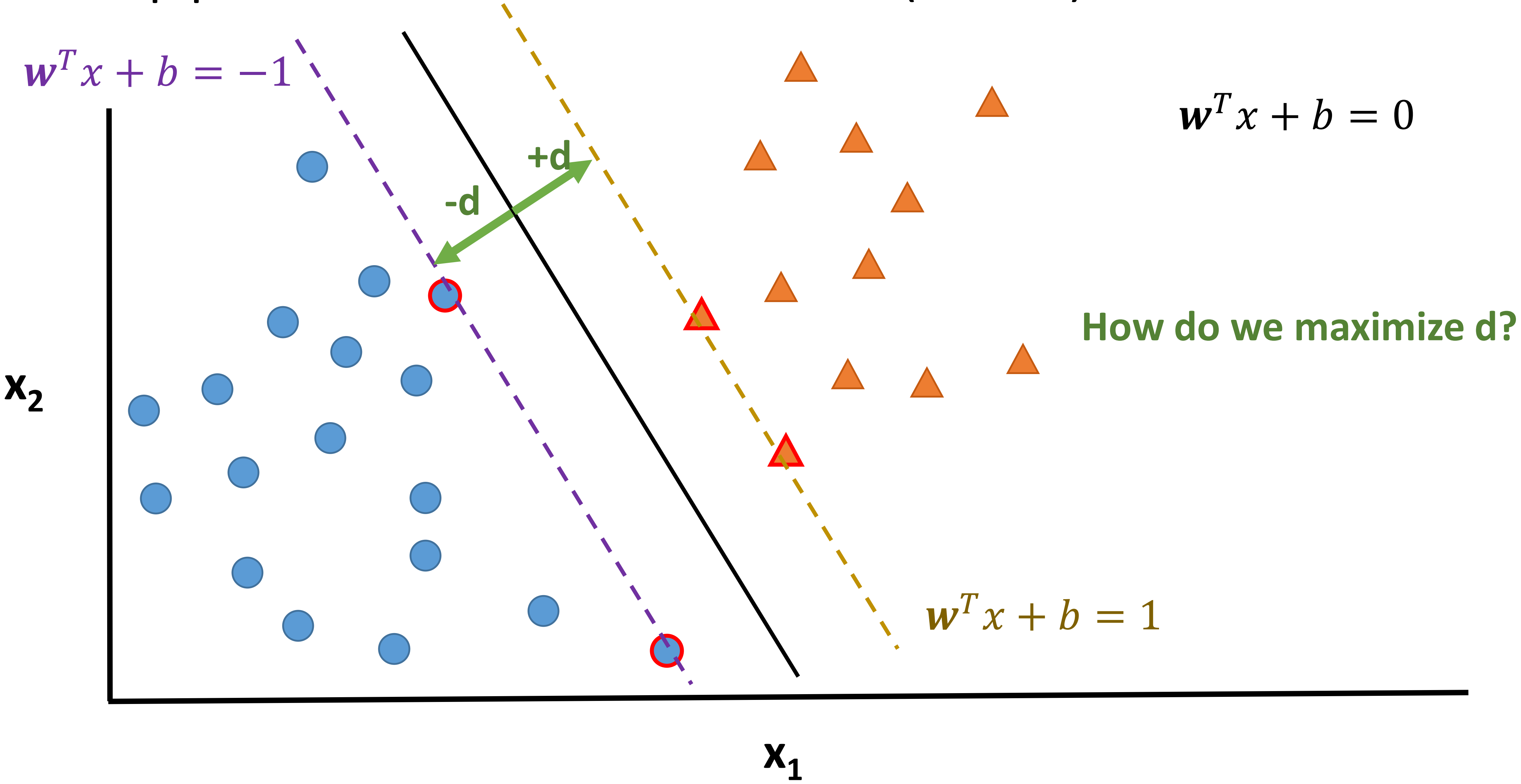
Support Vector Machines (SVM)



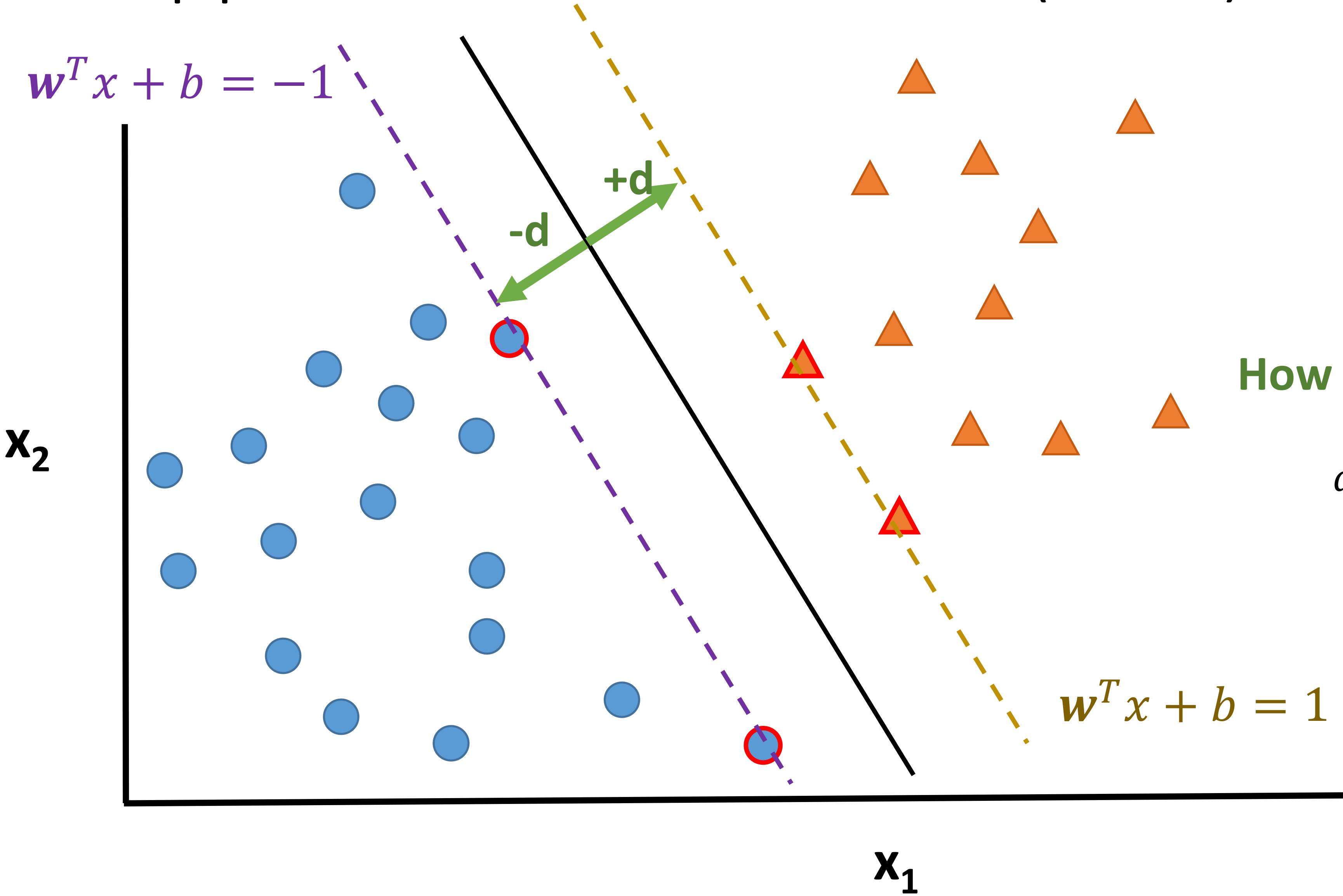
Support Vector Machines (SVM)



Support Vector Machines (SVM)



Support Vector Machines (SVM)



$$\boldsymbol{w}^T x + b = 0$$

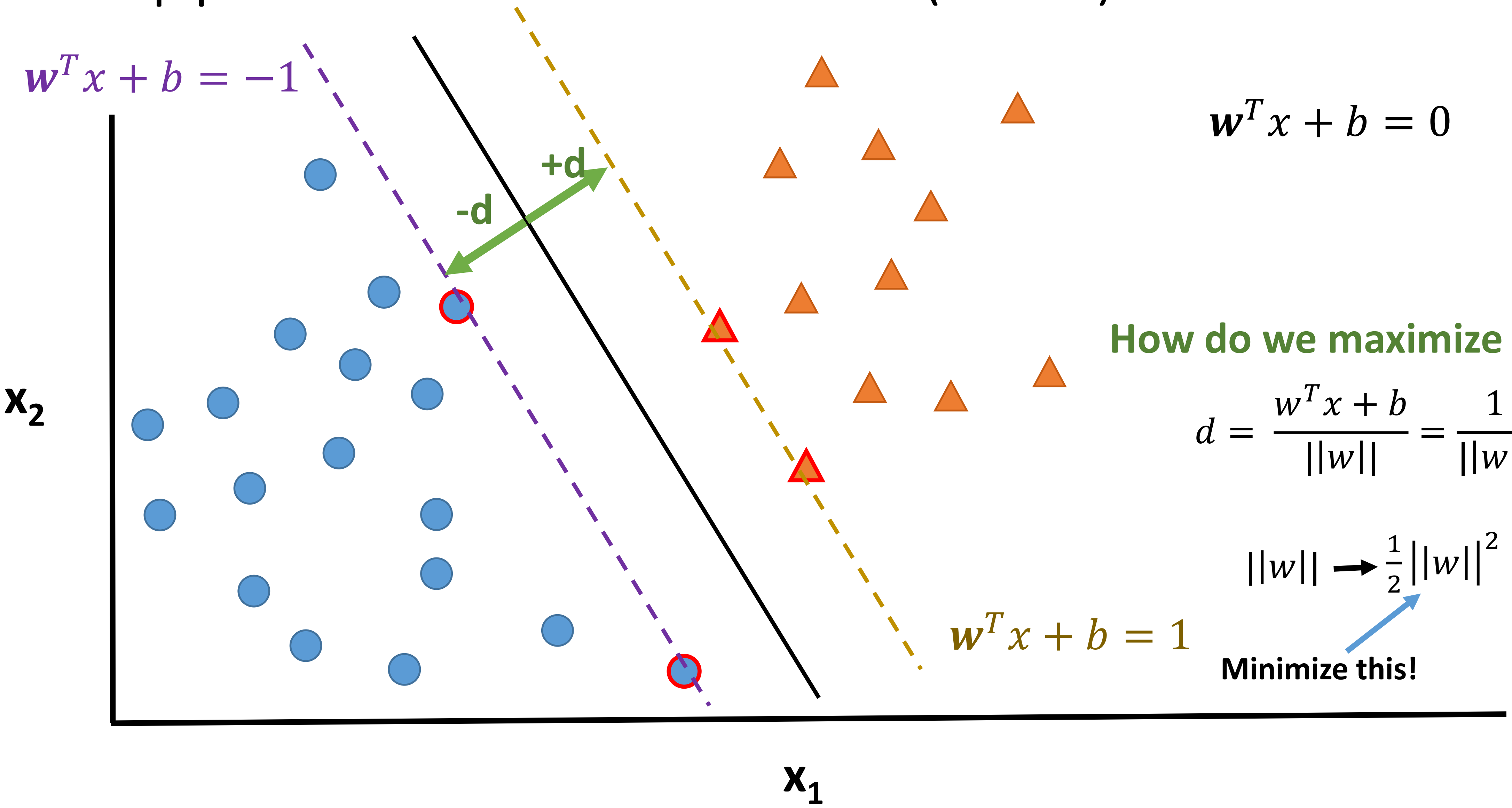
How do we maximize d?

$$d = \frac{w^T x + b}{||w||} = \frac{1}{||w||}$$

$$||w||$$

Minimize this!

Support Vector Machines (SVM)



$$\boldsymbol{w}^T x + b = 0$$

How do we maximize d?

$$d = \frac{w^T x + b}{||w||} = \frac{1}{||w||}$$

$$||w|| \longrightarrow \frac{1}{2}||w||^2$$

Minimize this!

SVM Initial Constraints

If our data point is labelled as y = 1

$$\left(w^T x^{(i)} + b\right) \geq 1$$

If y = -1

$$\left(w^T x^{(i)} + b\right) \leq -1$$

This can be succinctly summarized as:

$$y\left(w^T x^{(i)} + b\right) \geq 1$$

Hard Margin SVM

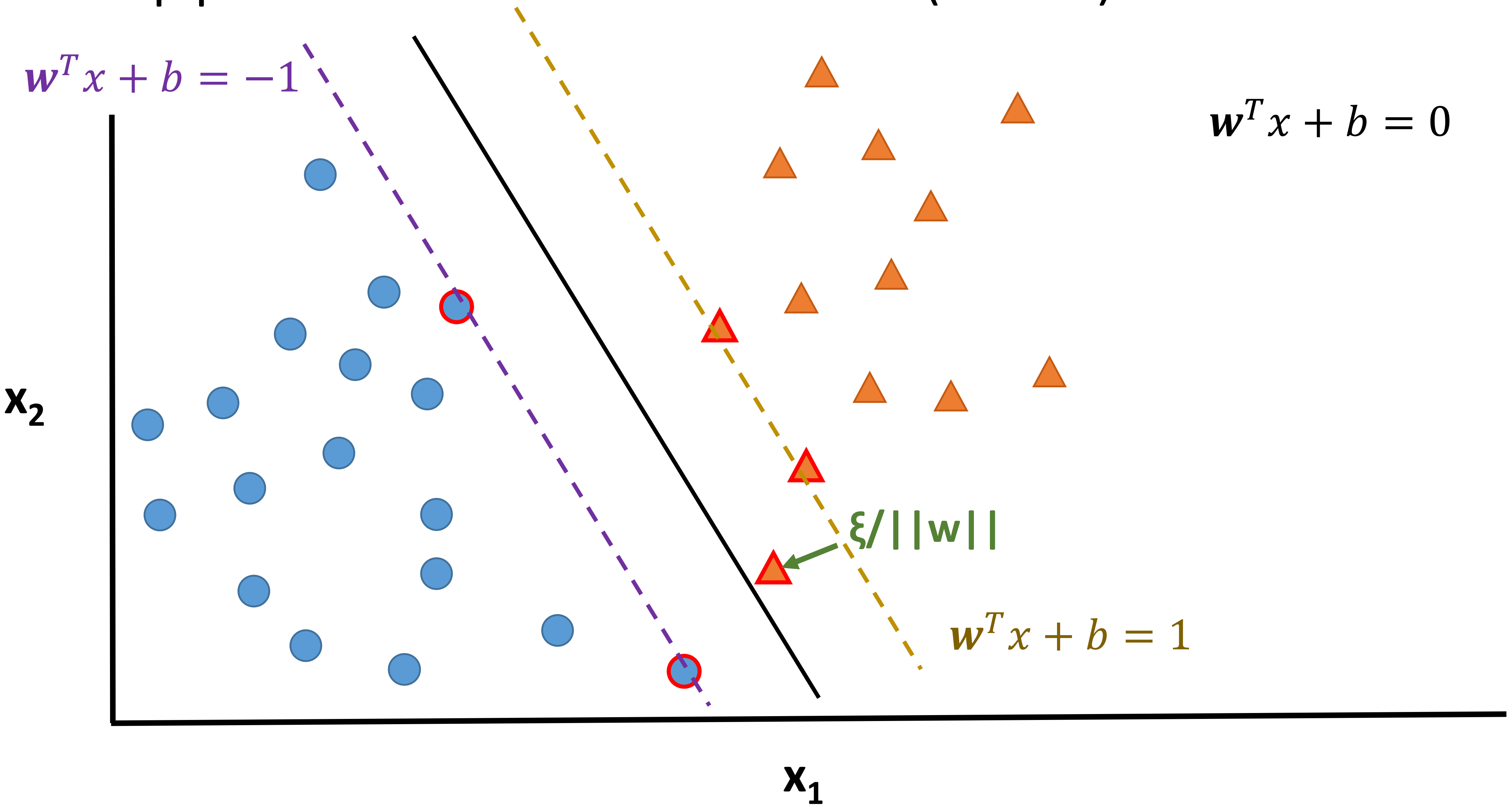
We're going to allow for some margin violation:

$$y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 - \xi_i$$

Soft Margin SVM

$$-y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi_i \leq 0$$

Support Vector Machines (SVM)



Optimization Objective

Minimize: $\frac{1}{2}||w||^2 + C \sum_i^m \xi_i$ ⟵ Penalizing Term

Subject to: $-y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi_i \leq 0$

$\xi_i \geq 0$

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment

Optimization Objective

Minimize $\frac{1}{2}\left|\left|w\right|\right|^2 + C\sum_i^m \xi_i$ s.t $-y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi_i \leq 0$ and $\xi_i \geq 0$

Need to use the Lagrangian

Using Lagrangian with inequalities as a constraint requires us to fulfill Karush-Khan-Tucker conditions…

Karush-Kahn-Tucker Conditions

There must exist a w* that solves the primal problem whereas α^* and b* are solutions to the dual problem. All three parameters must satisfy the following conditions:

1. $\frac{\partial}{\partial w_i}\mathcal{L}(w^*, \alpha^*, b^*) = 0$
2. $\frac{\partial}{\partial b_i}\mathcal{L}(w^*, \alpha^*, b^*) = 0$
3. $\alpha_i^* g_i(w^*) = 0$
4. $g_i(w^*) \leq 0$
5. $\alpha_i^* \geq 0$
6. $\xi_i \geq 0$
7. $r_i \xi_i = 0$

Example **x** *must lie directly on the functional margin* in order for $\boldsymbol{g_i}(\boldsymbol{w})$ **to equal 0**.

Here $\boldsymbol{\alpha}$ **> 0.**
This is our support vectors!

Constraint 3: If $\boldsymbol{\alpha}^*_i$ is a non-zero number, $\mathbf{g_i}$ must be **0**!

Recall: $g_i(\mathrm{w}) = -y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi_i \leq 0$

Optimization Objective

Minimize $\frac{1}{2}\left||w|\right|^2 + C\sum_i^m \xi_i$ subject to $-y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi_i \leq 0$ and $\xi_i \geq 0$

Need to use the Lagrangian

$$\mathcal{L}(x,\alpha) = f(x) + \sum_i \lambda_i \cdot h_i(x) \ldots$$

$$\mathcal{L}_{\mathcal{P}}(w,b,\alpha,\xi,r) = \frac{1}{2}\left||w|\right|^2 + C\sum_i \xi_i - \sum\nolimits_i \alpha_i\left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1 + \xi_i\right] - \sum_i r_i\xi_i$$

$$\min_{w,b} \mathcal{L}_{\mathcal{P}} = \frac{1}{2}\left||w|\right|^2 + C\sum_{i=1}^m \xi_i + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y^{(i)}\left[\left(w^T x^{(i)} + b\right) + \xi_i\right] - \sum_{i=1}^m r_i\xi_i$$

Optimization Objective

Minimize $\frac{1}{2}||w||^2 + C\sum_i^m \xi_i$ subject to $-y^{(i)}\left(w^T x^{(i)} + b\right) + 1 - \xi \leq 0$ and $\xi_i \geq 0$

Need to use the Lagrangian

$$\mathcal{L}(x, \alpha) = f(x) + \sum_i \lambda_i \cdot h_i(x) \ldots$$

$$\mathcal{L}_{\mathcal{P}}(w, b, \alpha, \xi, r) = \frac{1}{2}||w||^2 + C\sum_i \xi_i - \sum_i \alpha_i\left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1 + \xi_i\right] - \sum_i r_i \xi_i$$

$$\min_{w,b} \mathcal{L}_{\mathcal{P}} = \frac{1}{2}||w||^2 + C\sum_{i=1}^m \xi_i + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y^{(i)}\left[\left(w^T x^{(i)} + b\right) + \xi_i\right] - \sum_{i=1}^m r_i \xi_i$$

Minimization function **Margin Constraint** **Error Constraint**

The problem with the Primal

$$\mathcal{L}_{\mathcal{P}} = \frac{1}{2}\left|\left|w\right|\right|^2 + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i - \sum_{i=1}^{m}\alpha_i[y^{(i)}\left(w^T x^{(i)} + b\right) + \xi_i] - \sum_{i=1}^{m} r_i\xi_i$$

We could just optimize the primal equation and be finished with SVM optimization…
BUT!!!
You would miss out on one of the most important aspects of the SVM
The Kernel Trick…

We need the **Lagrangian Dual!**

Why bother with the dual problem when fitting SVM?

23

18

Given the data points $x_1, \ldots, x_n \in \mathbb{R}^d$ and labels $y_1, \ldots, y_n \in \{-1, 1\}$, the hard margin SVM primal problem is

$$\text{minimize}_{w,w_0} \quad \frac{1}{2} w^T w$$

$$\text{s.t.} \quad \forall i : y_i(w^T x_i + w_0) \geq 1$$

which is a quadratic program with $d+1$ variables to be optimized for and i constraints. The dual

9 Short answer is kernels. Long answer is keeerneeels (-; – mbq ♦ Dec 1 '11 at 0:26

http://stats.stackexchange.com/questions/19181/why-bother-with-the-dual-problem-when-fitting-svm

Finding the Dual

$$\mathcal{L}_{\mathcal{P}} = \frac{1}{2}||w||^2 + \underline{C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i - \sum_{i=1}^{m}\alpha_i\left[y^{(i)}\left(w^T x^{(i)} + b\right) + \xi_i\right]} - \sum_{i=1}^{m} r_i\xi_i$$

First, compute minimal **w**, **b** and $\boldsymbol{\xi}$.

$$\nabla_w \mathcal{L}(w, b, \alpha) = w - \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)} = 0$$

Substitute in

Can solve for w!

$$w = \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)}$$

New constraint

$$\frac{\partial}{\partial b}\mathcal{L}(w, b, \alpha) = \sum_{i=1}^{m}\alpha_i y^{(i)} = 0$$

$$0 \le \alpha_i \le C$$

New constraint

$$\frac{\partial}{\partial \xi}\mathcal{L}(w, b, \alpha) = C - \alpha_i - r_i = 0 \longrightarrow \alpha_i = C - r_i$$

Lagrangian Dual

$$\mathcal{L}_{\mathcal{P}} = \frac{1}{2}\left||w|\right|^2 + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i - \sum_{i=1}^{m}\alpha_i y^{(i)}\left[\left(w^T x^{(i)} + b\right) + \xi_i\right] - \sum_{i=1}^{m} r_i\xi_i$$

$$w = \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)}$$

$$\mathcal{L}_{\mathcal{D}} = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)} y^{(j)}\left\langle x^{(i)} \cdot x^{(j)}\right\rangle$$

s.t

∀i : $\sum_{i=1}^{m}\alpha_i y^{(i)} = 0, \xi \geq 0$ and $0 \leq \alpha_i \leq C$(from KKT and prev. derivation)

Lagrangian Dual

$$\max_{\alpha} \mathcal{L}_{\mathcal{D}} = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} \left\langle \boldsymbol{x}^{(\boldsymbol{i})} \cdot \boldsymbol{x}^{(\boldsymbol{j})} \right\rangle$$

- S.T: ∀i : $\sum_{i=1}^{m} \alpha_i y^{(i)} = 0, \xi \geq 0, 0 \leq \alpha_i \leq C$ and KKT conditions
- Compute **inner product of $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$**
  - No longer rely on **w** or **b**
  - Can replace with *K(x,z)* for **kernels**
- Can solve for $\boldsymbol{\alpha}$ **explicitly**
  - All values not on functional margin have α = 0, more efficient!

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic SVM Gradient Descent (extra)

Sequential Minimal Optimization

$$\max_{\alpha} \mathcal{L}_{\mathcal{D}} = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} \langle \boldsymbol{x}^{(\boldsymbol{i})} \cdot \boldsymbol{x}^{(\boldsymbol{j})} \rangle$$

- Ideally, hold all α but α_j constant and optimize but:

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \longrightarrow \alpha_1 = -y^1 \sum_{i=2}^{m} \alpha_i y^{(i)}$$

- **Solution:** Change two α at a time!

Platt 1998

Sequential Minimal Optimization Algorithm

Initialize all α to 0

Repeat {

1. Select α_j that violates KKT and additional α_k to update
2. Reoptimize $\mathcal{L}_{\mathcal{D}}(\alpha)$ with respect to two α's while holding all other α constant and compute w,*b*

}

In order to keep $\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$ true:

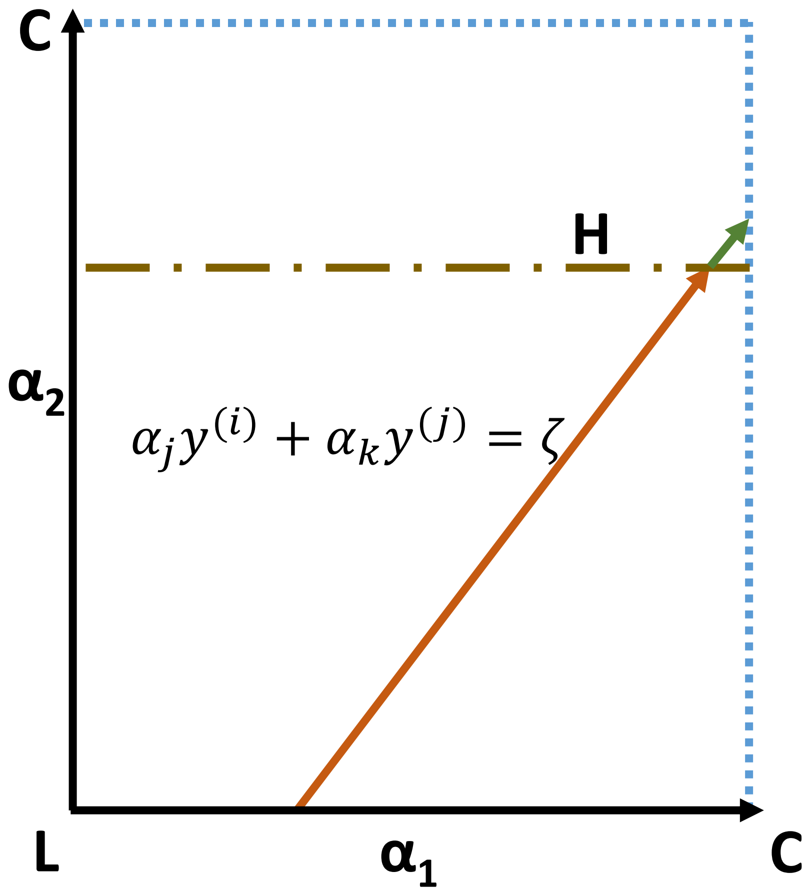
$$\alpha_j y^{(i)} + \alpha_k y^{(j)} = \zeta$$

Even after update, **linear combination must remain constant**!

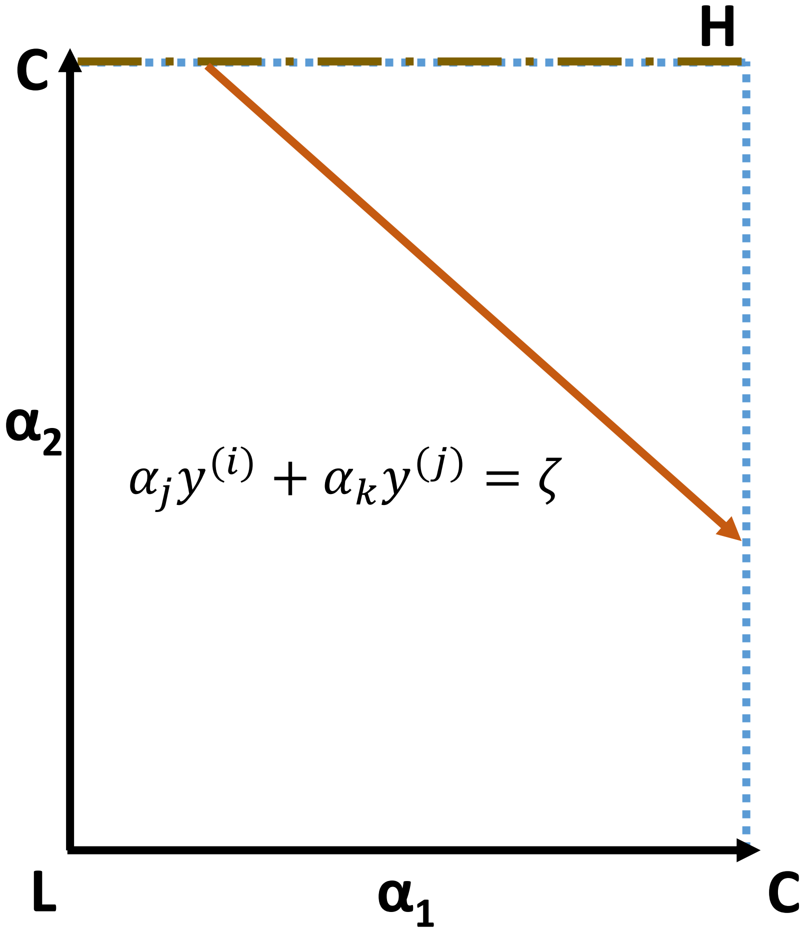
Platt 1998

Sequential Minimal Optimization

- $0 \leq \alpha \leq C$ from constraint
- Line given due to linear constraint
- Clip values if they exceed C, H or L



If $y^{(j)} \neq y^{(k)} then\ \alpha_j - \alpha_k = \zeta$



If $y^{(j)} = y^{(k)} then\ \alpha_j + \alpha_k = \zeta$

Sequential Minimal Optimization

Can analytically solve for new α, w and b for each update step (closed form computation)

$\alpha_k^{new} = \alpha_k^{old} + \frac{y^{(k)}(E_j^{old} - E_k^{old})}{\eta}$ where η is second derivative of Dual with respect to α_2.

$$\alpha_j^{new} = \alpha_j^{old} + y^{(k)}y^{(j)}(\alpha_k^{old} - \alpha_k^{new,clipped})$$

$$b_1 = E_1 + y^{(j)}\left(\alpha_j^{new} - \alpha_j^{old}\right)K\left(x^{(j)}, x^{(j)}\right) + y^{(k)}\left(\alpha_k^{new,clipped} - \alpha_k^{old}\right)K(x^{(j)}, x^{(k)})$$

$$b_2 = E_2 + y^{(j)}\left(\alpha_j^{new} - \alpha_j^{old}\right)K\left(x^{(j)}, x^{(k)}\right) + y^{(k)}\left(\alpha_k^{new,clipped} - \alpha_k^{old}\right)K(x^{(k)}, x^{(k)})$$

$$b = \frac{b_1 + b_2}{2}$$

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}$$

Platt 1998

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic SVM Gradient Descent (extra)

Kernels

Recall: $\max_{\alpha} \mathcal{L}_{\mathcal{D}} = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} \langle \boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)} \rangle$

We can generalize this using a kernel function $\mathbf{K}(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$!

$$\mathrm{K}\left(x^{(i)}, x^{(j)}\right) = \phi(x^{(i)})^T \phi(x^{(j)})$$

Where ϕ is a feature mapping function.

Turns out that we don't need to explicitly compute $\phi(x)$ due to **Kernel Trick!**

Kernel Trick

Suppose $\mathrm{K}(x, z) = \left(x^T z\right)^2$

$$\mathrm{K}(x, z) = \left(\sum_{i=1}^{m} x^{(i)} z^{(i)}\right)\left(\sum_{j=1}^{m} x^{(j)} z^{(j)}\right)$$

$$K(x, z) = \sum_{i=1}^{m}\sum_{j=1}^{m} x^{(i)} z^{(j)} x^{(i)} z^{(j)}$$

$$K(x, z) = \sum_{i,j=1}^{m} (x^{(i)} x^{(j)})(z^{(i)} z^{(j)})$$

Representing each feature: **O(n²)**

Kernel trick: **O(n)**

Types of Kernels

Linear Kernel

$$\mathrm{K}(x, z) = x^T z$$

Gaussian Kernal (Radial Basis Function)

$$K(x, z) = \exp\left(\frac{\left||x - z|\right|^2}{2\sigma^2}\right)$$

Polynomial Kernel

$$K(x, z) = \left(x^T z + c\right)^d$$

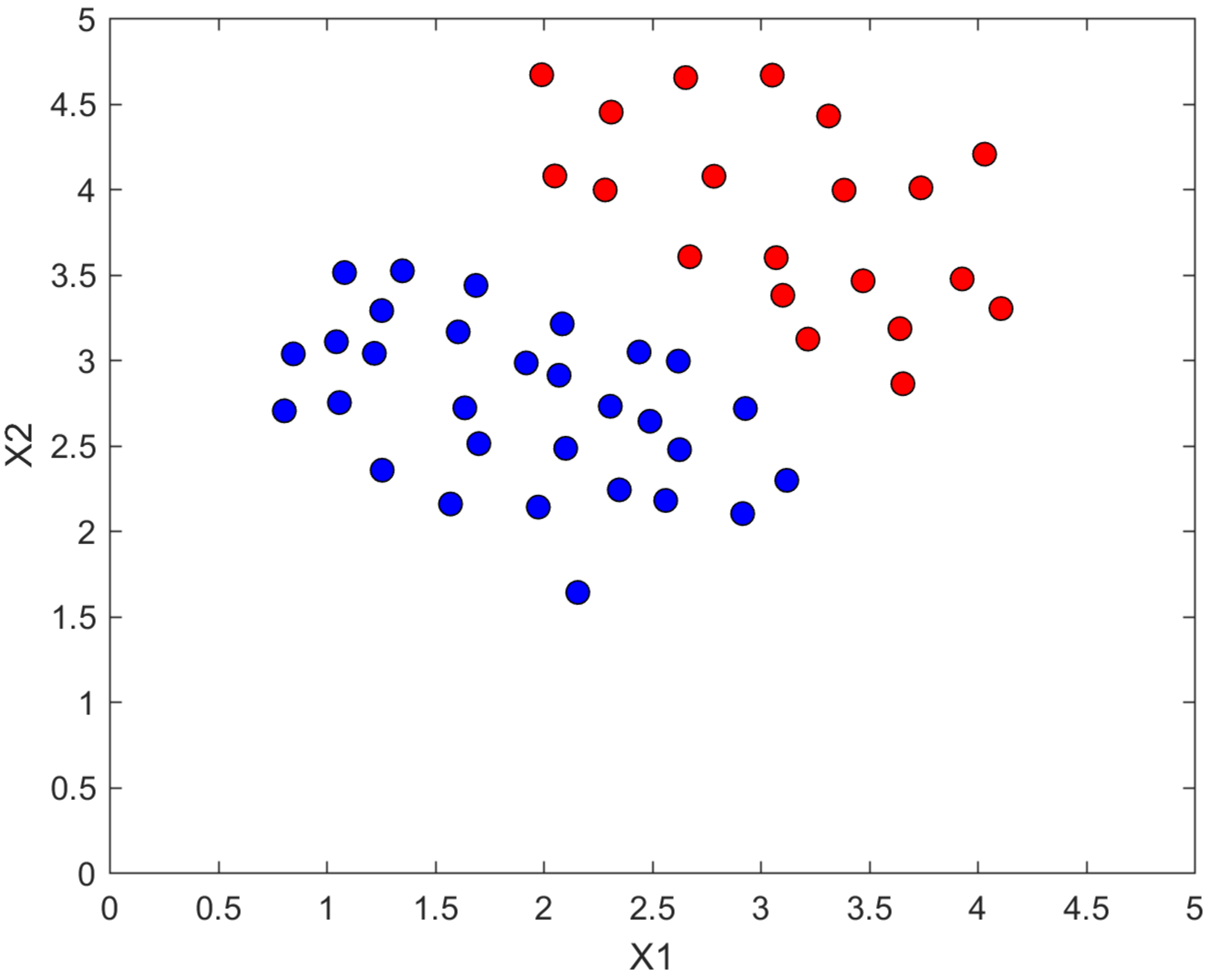
$$\max_{\alpha} \mathcal{L}_{\mathcal{D}} = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} \boldsymbol{K}\left(\boldsymbol{x}^{(\boldsymbol{i})}, \boldsymbol{x}^{(\boldsymbol{j})}\right)$$

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic SVM Gradient Descent (extra)

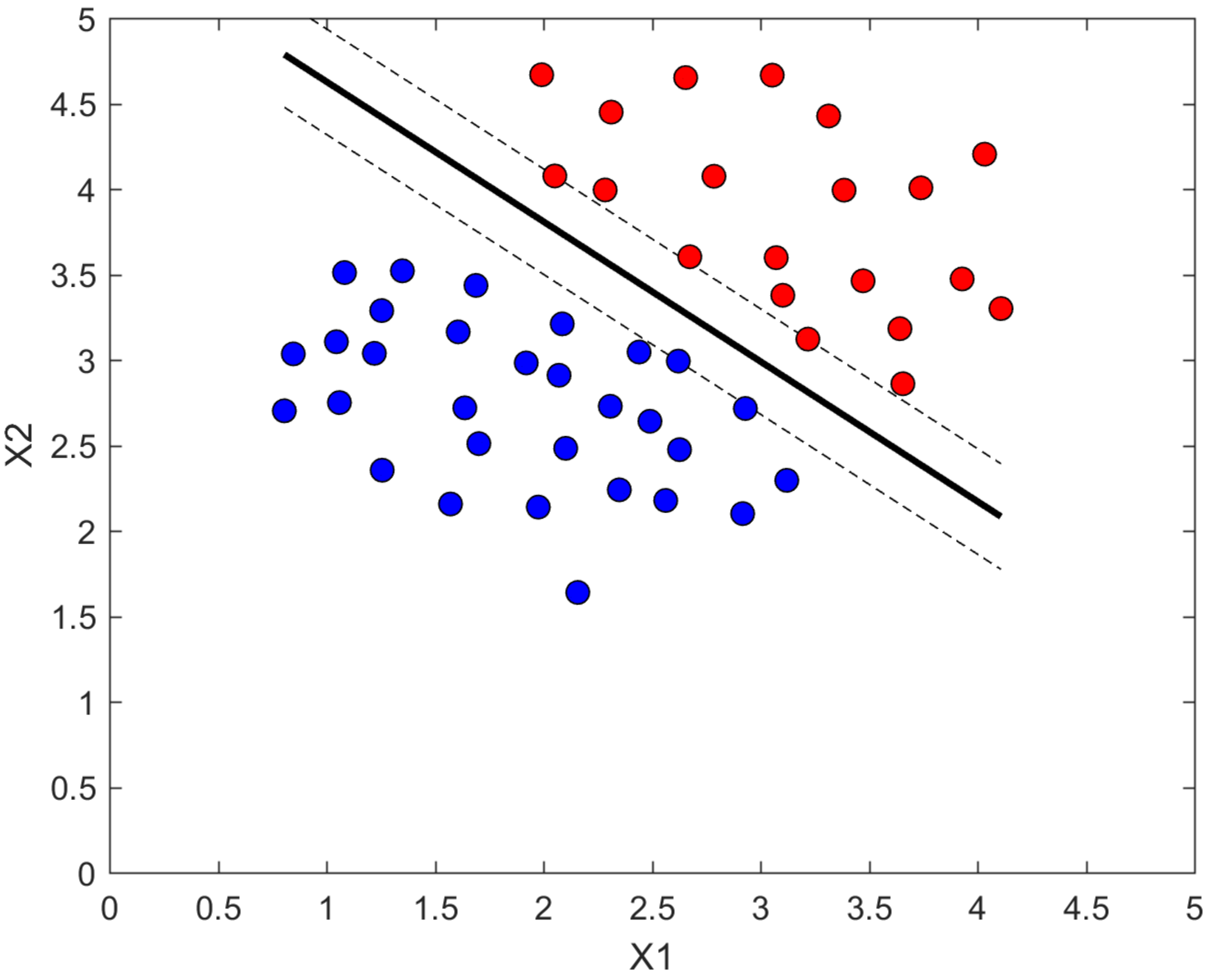
Example 1: Linear Kernel



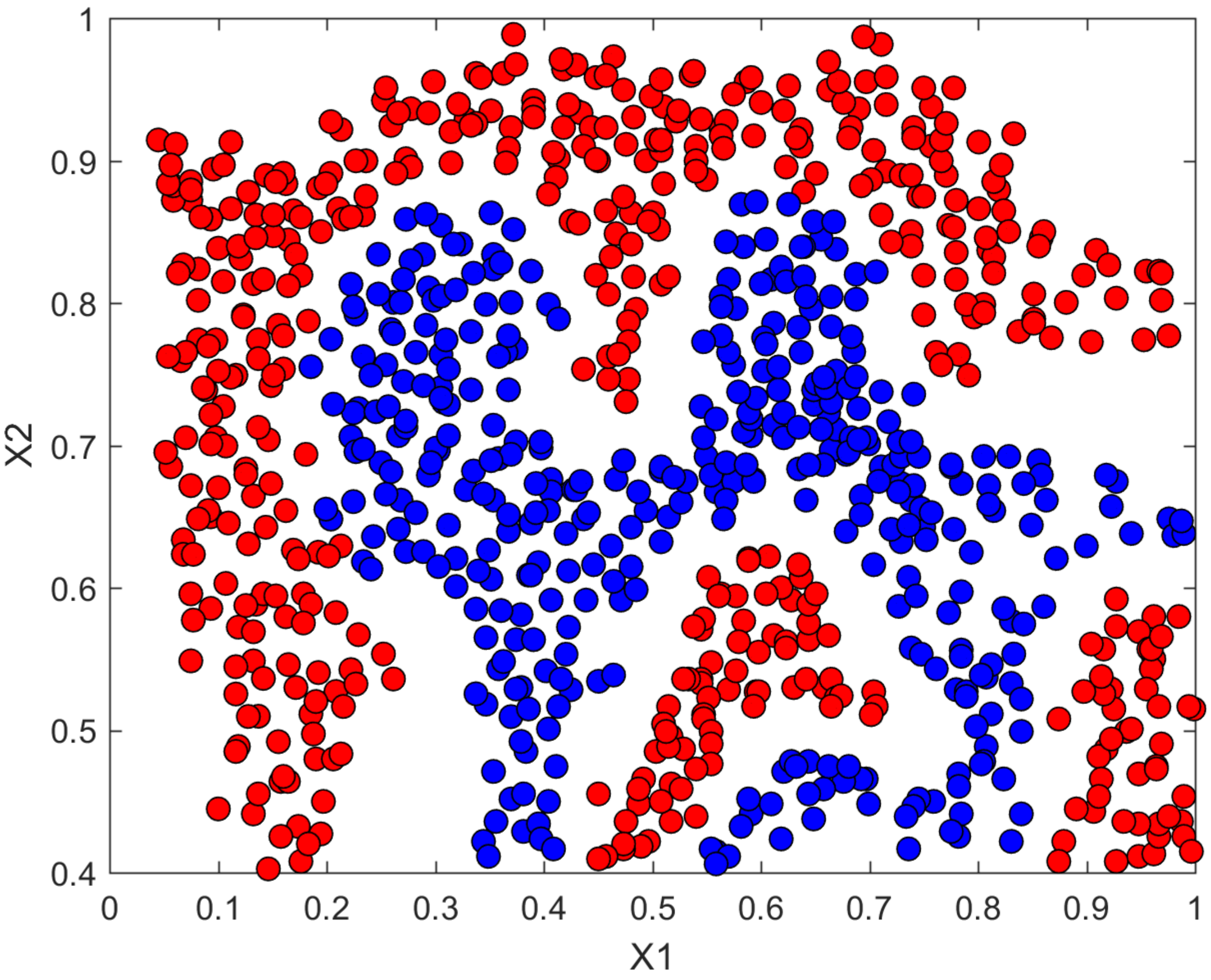
Example 1: Linear Kernel

In MATLAB:
I used *svmtrain:*
model = svmTrain(X, y, C, @linearKernel, 1e-3, 500);

- 500 max iterations
- C = 1000
- KKT *tol* = 1e-3



Example 2: Gaussian Kernel (RBF)

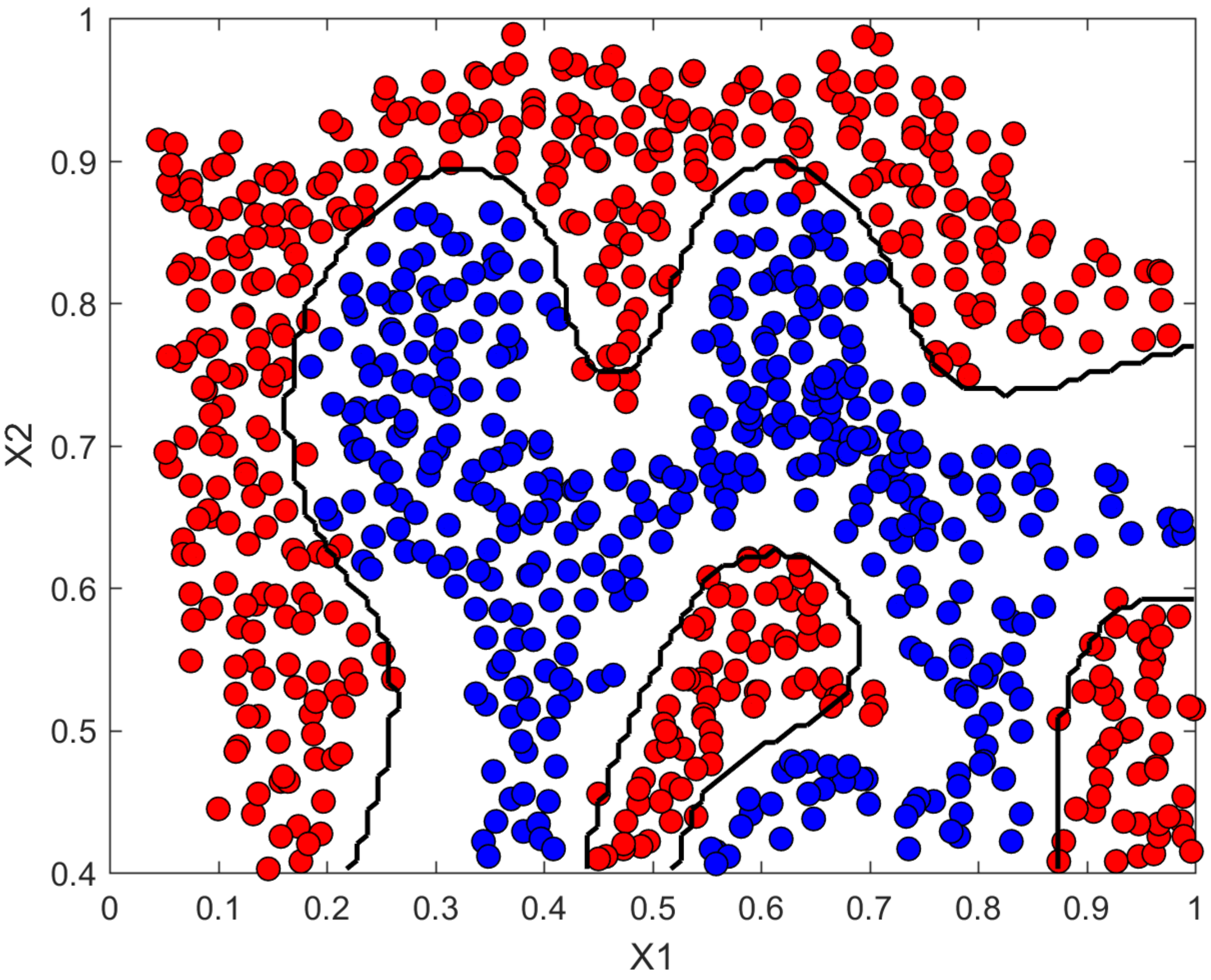


Example 2: Gaussian Kernel (RBF)

In MATLAB
Gaussian Kernel:
gSim = exp(-1*((x1-x2)'*(x1-x2))/(2*sigma^2));

model= svmTrain(X, y, C, @(x1, x2) gaussianKernel(x1, x2, sigma), 0.1, 300);

- 300 max iterations
- C = 1
- KKT *tol* = 0.1

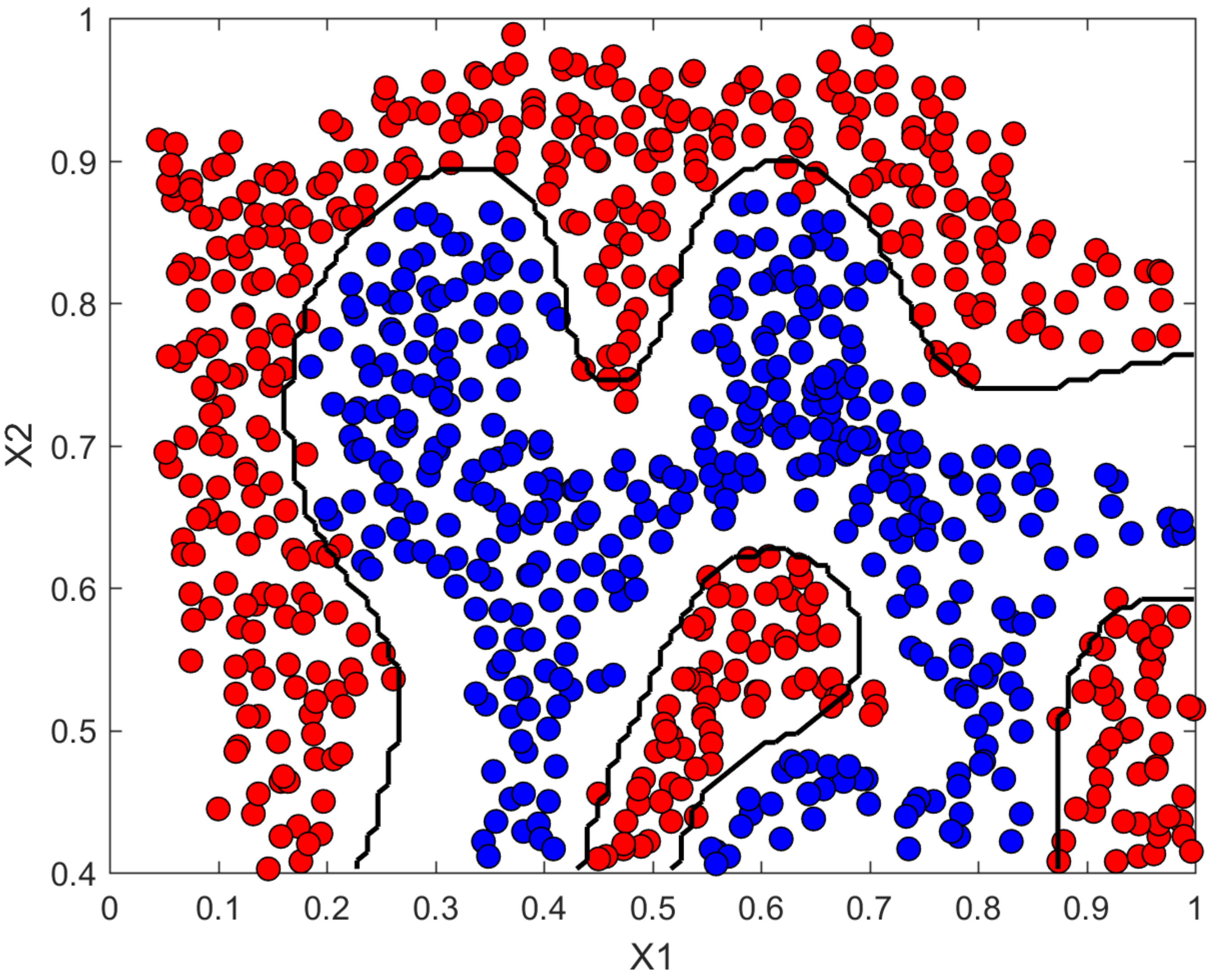


Example 2: Gaussian Kernel (RBF)

In MATLAB
Gaussian Kernel:
gSim = exp(-1*((x1-x2)'*(x1-x2))/(2*sigma^2));

model= svmTrain(X, y, C, @(x1, x2) gaussianKernel(x1, x2, sigma), 0.001, 300);

- 300 max iterations
- C = 1
- **KKT *tol* = 0.001**

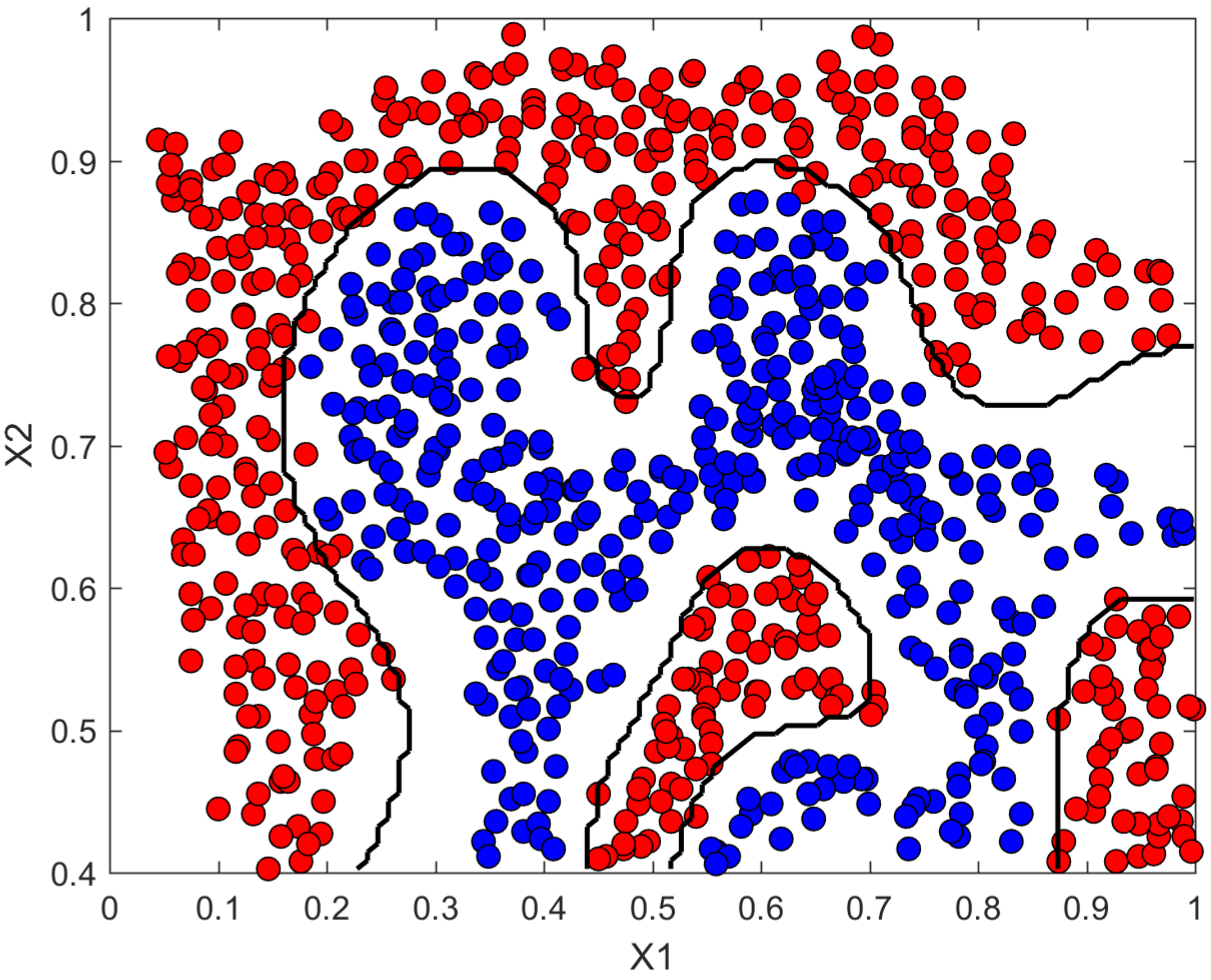


Example 2: Gaussian Kernel (RBF)

In MATLAB
Gaussian Kernel:
gSim = exp(-1*((x1-x2)'*(x1-x2))/(2*sigma^2));

model= svmTrain(X, y, C, @(x1, x2) gaussianKernel(x1, x2, sigma), 0.001, 300);

- 300 max iterations
- **C = 10**
- KKT *tol* = 0.001



Resources

- **http://research.microsoft.com/pubs/69644/tr-98-14.pdf**
- ftp://www.ai.mit.edu/pub/users/tlp/projects/svm/svm-smo/smo.pdf
- http://www.cs.toronto.edu/~urtasun/courses/CSC2515/09svm-2515.pdf
- http://www.pstat.ucsb.edu/student%20seminar%20doc/svm2.pdf
- http://www.ics.uci.edu/~dramanan/teaching/ics273a_winter08/lectures/lecture11.pdf
- http://stats.stackexchange.com/questions/19181/why-bother-with-the-dual-problem-when-fitting-svm
- **http://cs229.stanford.edu/notes/cs229-notes3.pdf**
- http://www.svms.org/tutorials/Berwick2003.pdf
- http://www.med.nyu.edu/chibi/sites/default/files/chibi/Final.pdf
- https://share.coursera.org/wiki/index.php/ML:Main
- https://www.youtube.com/watch?v=vqoVIchkM7I

Outline

- Main goal: To understand how support vector machines (SVMs) perform optimal classification for labelled data sets, also a quick peek at the implementational level.

1. What is a support vector machine?
2. Formulating the SVM problem
3. Optimization using Sequential Minimal Optimization
4. Use of Kernels for non-linear classification
5. Implementation of SVM in MATLAB environment
6. Stochastic SVM Gradient Descent (extra)

Hinge-Loss Primal SVM

- **Recall:** Original Problem

$\frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i$ S.T: $y^{(i)}\left(w^T x^{(i)} + b\right) - 1 + \xi \geq 0$

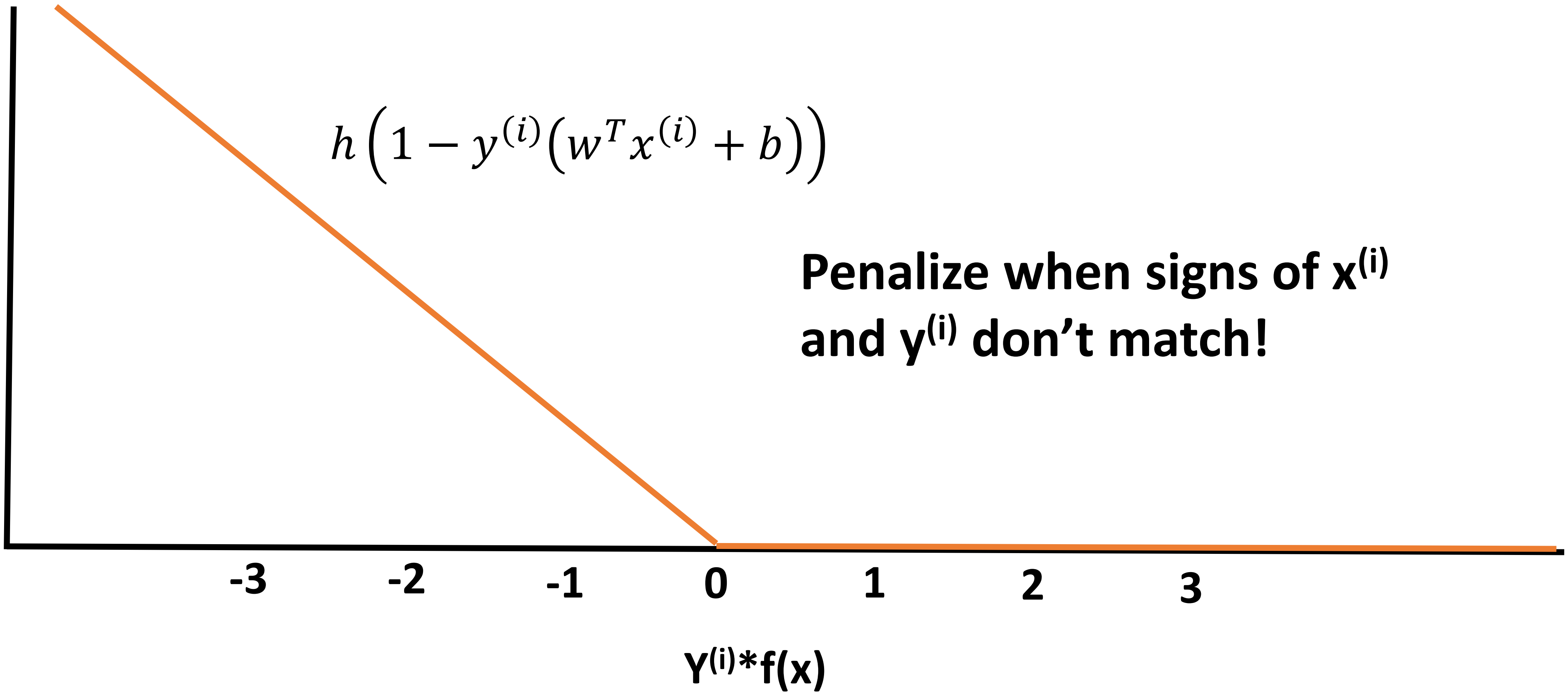
Re-write in terms of hinge-loss function

$$\mathcal{L}_{\mathcal{P}} = \frac{\lambda}{2} w^T w + \frac{1}{n} \sum_{i=1}^{m} h\left(1 - y^{(i)}\left(w^T x^{(i)} + b\right)\right)$$

Regularization

Hinge-Loss Cost Function

Hinge Loss Function



Penalize when signs of $x^{(i)}$ and $y^{(i)}$ don't match!

Hinge-Loss Primal SVM

Hinge-Loss Primal

$$\mathcal{L}_{\mathcal{P}} = \frac{\lambda}{2} w^T w + \frac{1}{n} \sum_{i=1}^{m} h\left(1 - y^{(i)}\left(w^T x^{(i)} + b\right)\right)$$

Compute gradient (**sub-gradient**)

$$\nabla_w = \lambda w - \frac{1}{n} \sum_{i=1}^{n} y^{(i)} x^{(i)} L\left(y^{(i)}\left(w^T x^{(i)} + b\right) < 1\right)$$

Indicator function
$L(y^{(i)}f(x)) = 0$ if classified correctly
$L(y^{(i)}f(x)) =$ hinge slope if classified incorrectly

Stochastic Gradient Descent

We implement random sampling here to pick out data points:

$$\lambda w \; - \; \mathbb{E}_{i \in \mathbb{U}}\left[y^{(i)} x^{(i)} l\left(y^{(i)}\left(w^T x^{(i)} + b\right) < 1\right)\right]$$

Use this to compute update rule:

$$w_t \coloneqq \; w_{t-1} + \frac{1}{t}\left(y^{(i)} x^{(i)} l\left(y^{(i)}\left(x^{(i)T} w_{t-1} + b\right) < 1\right) - \lambda w_{t-1}\right)$$

Here *i* is sampled from a uniform distribution.

Shalev-Shwartz et al. 2007