Advanced Introduction to Deep Learning
Lecture 4: Kernel Methods

Han Zhao
han.zhao@cs.cmu.edu
Machine Learning Department, Carnegie Mellon University

July. 23rd, 2019

Lecture 4: Kernel Methods

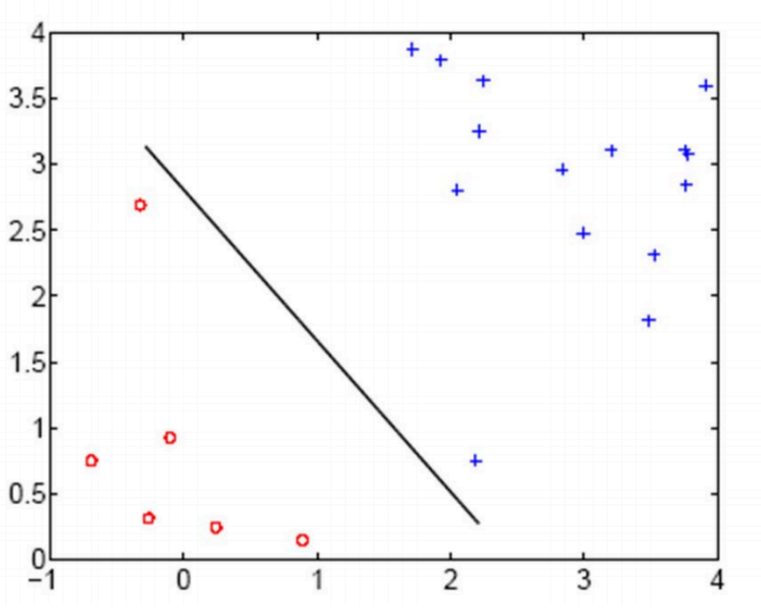
Overview:

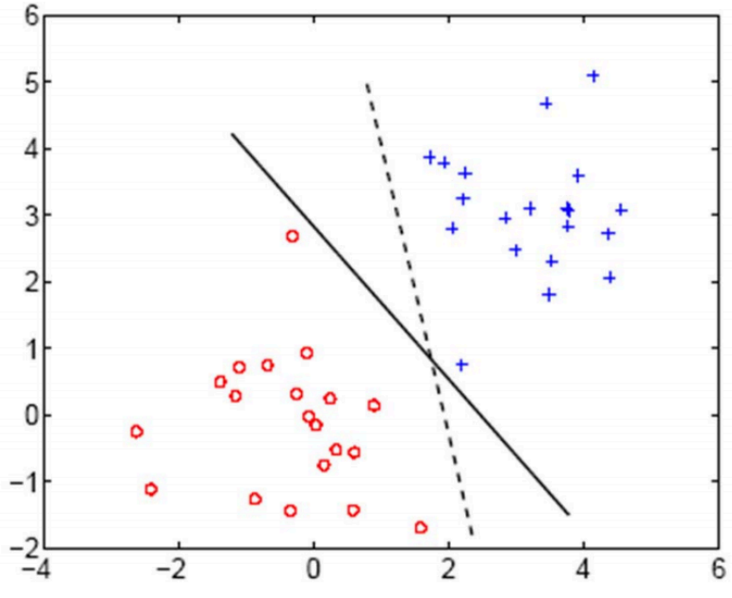
- Kernel Methods
  - Linear Support Vector Machines
  - Nonlinear Support Vector Machines

Kernel Methods: Linear SVM

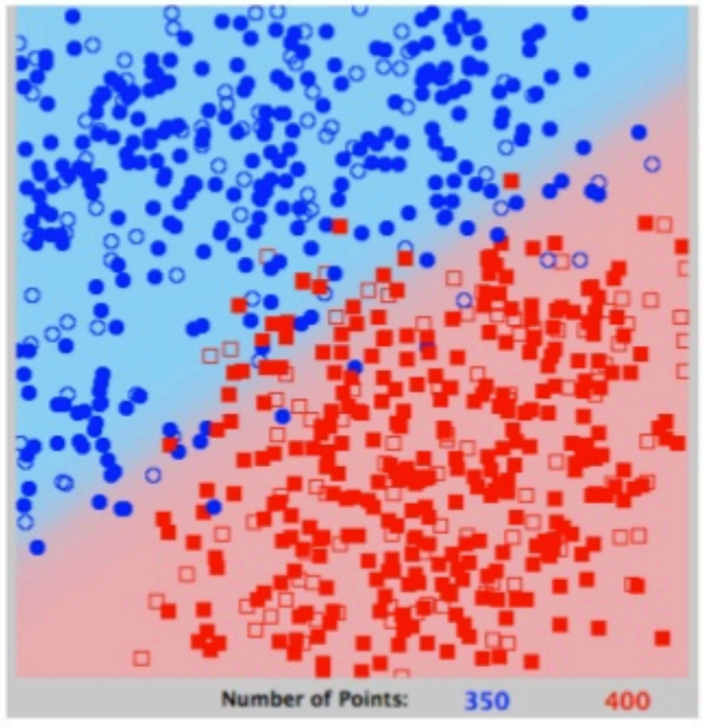
Recall in the last lectures we have learned a linear model for classification (Logistic Regression)

- If there are many equally good linear decision boundaries, which one should we prefer?
- Clearly, not all boundaries are equal



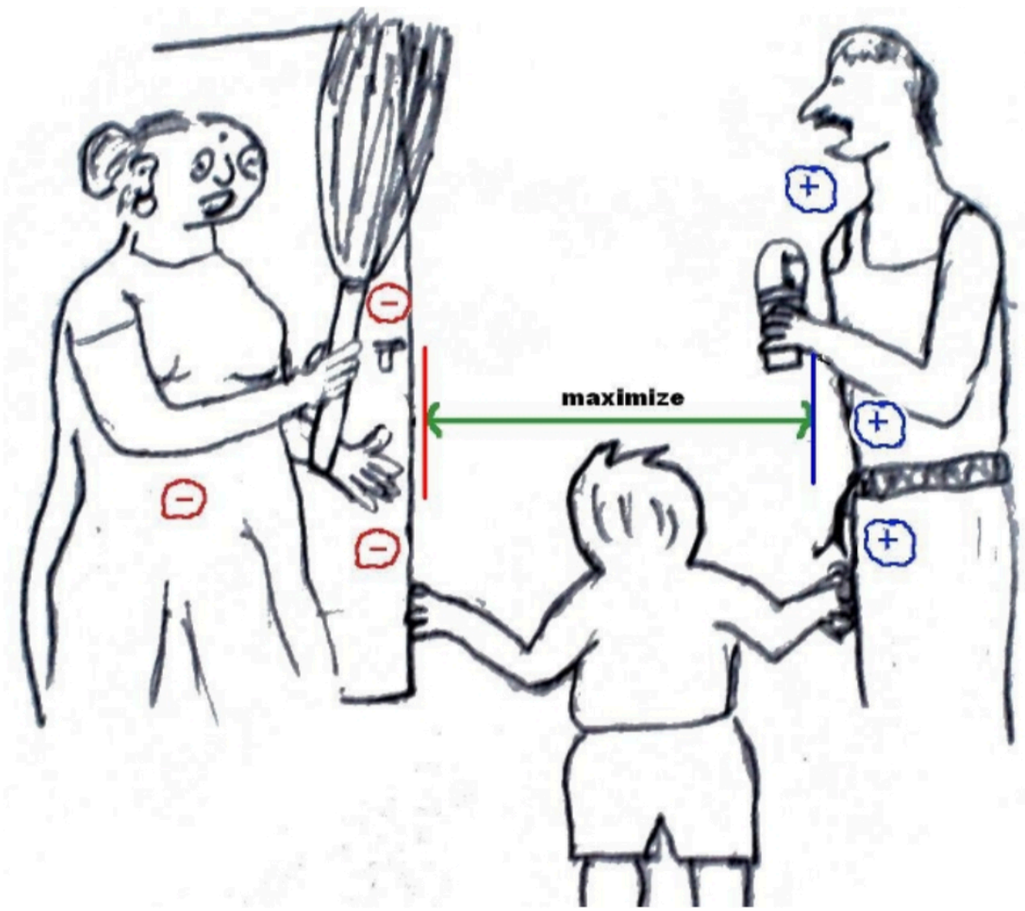


Decision surface is linear



Kernel Methods: Linear SVM

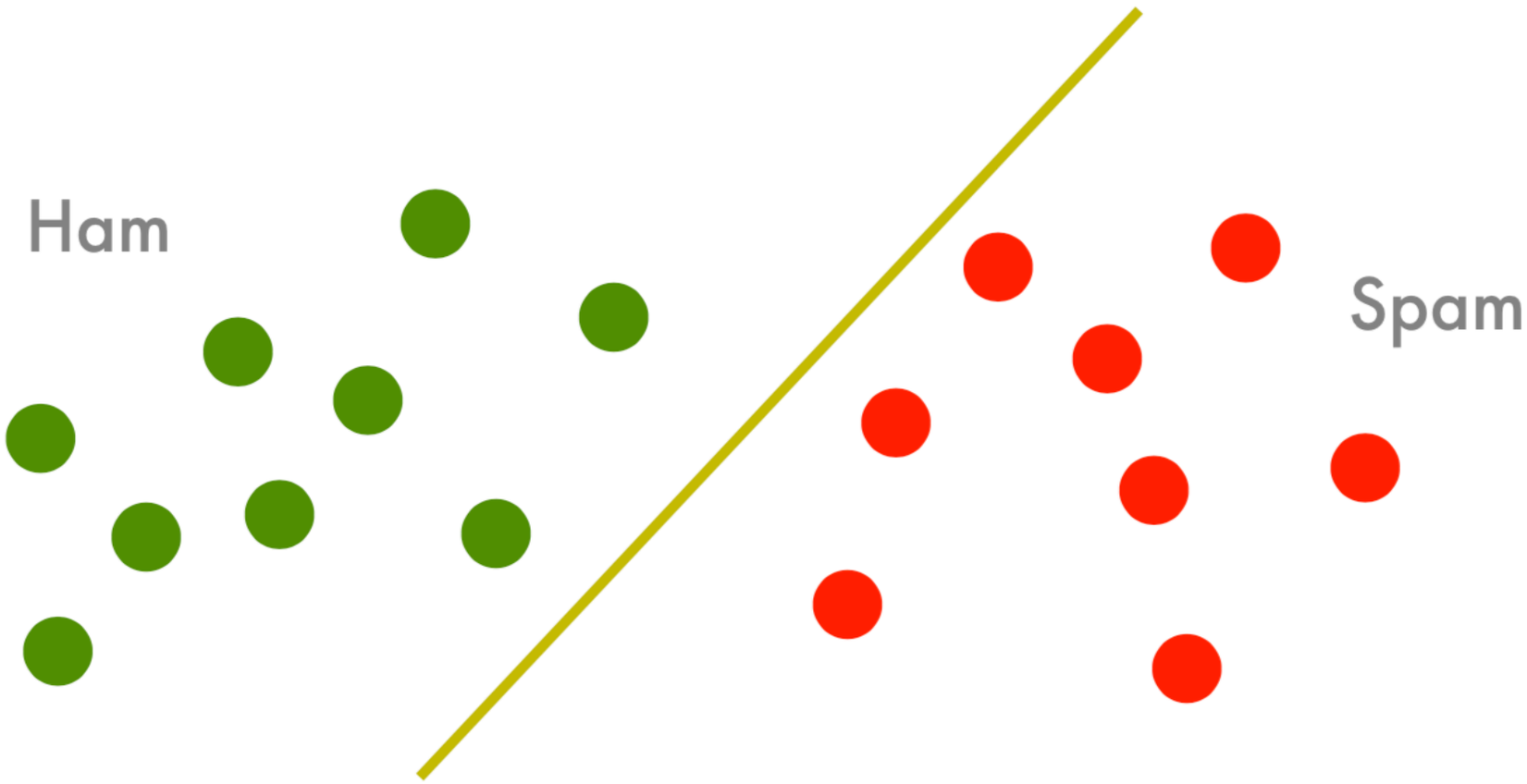
Support Vector Machines:



Key idea: find the linear classifier that maximizes the margin!

Kernel Methods: Linear SVM

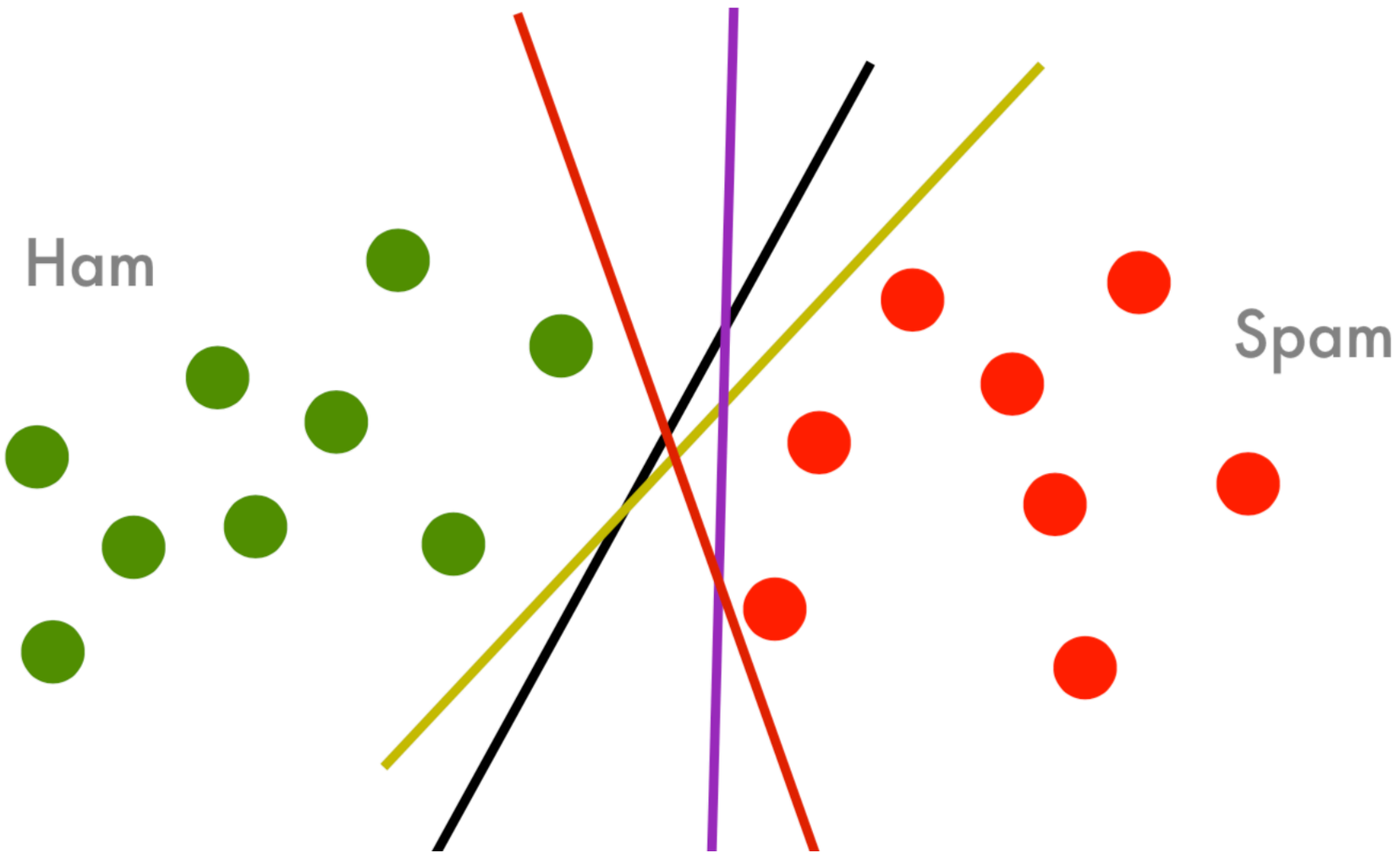
Support Vector Machines:



Key idea: find the linear classifier that maximizes the margin!

Kernel Methods: Linear SVM

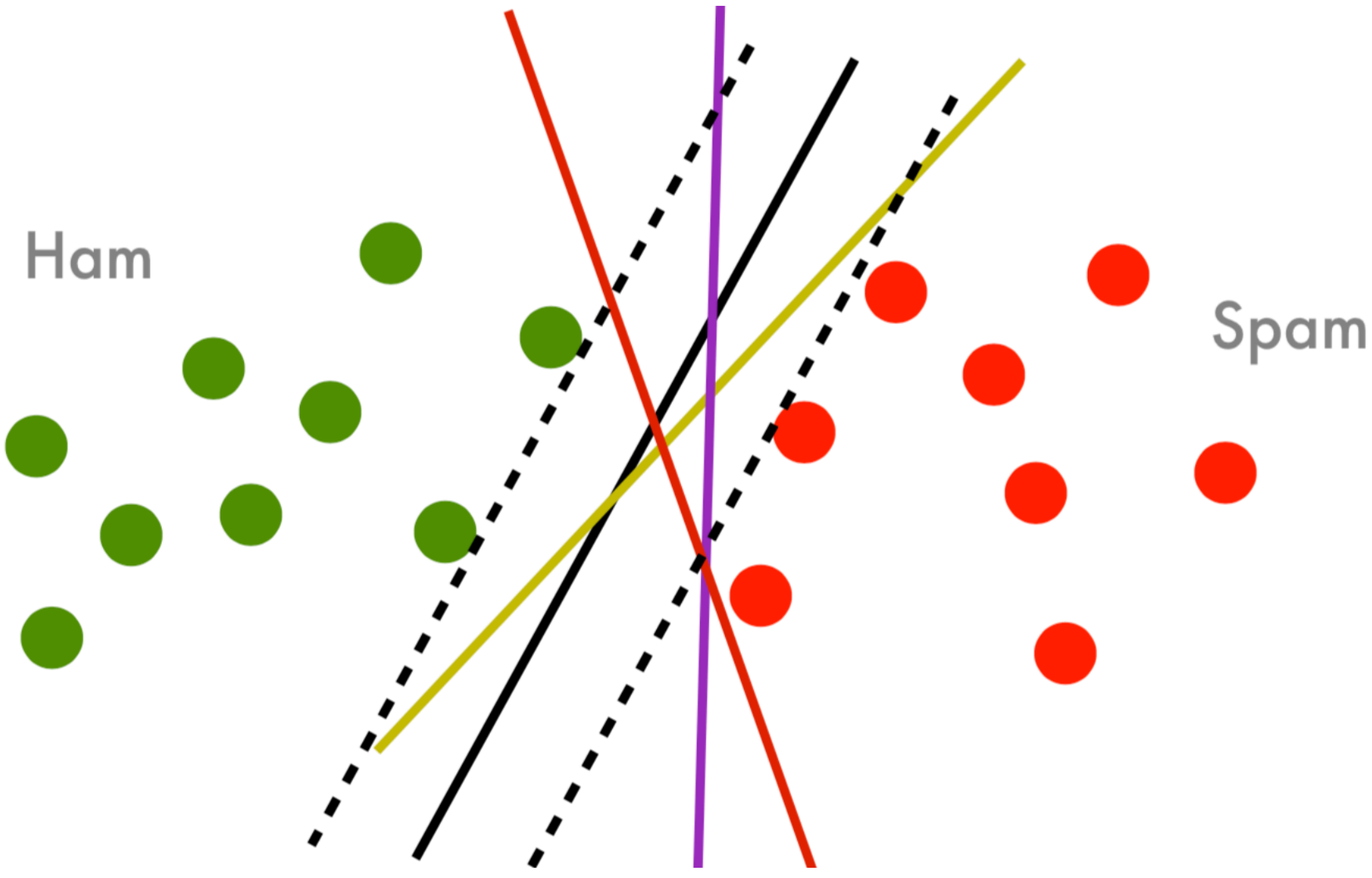
Support Vector Machines:



Key idea: find the linear classifier that maximizes the margin!

Kernel Methods: Linear SVM

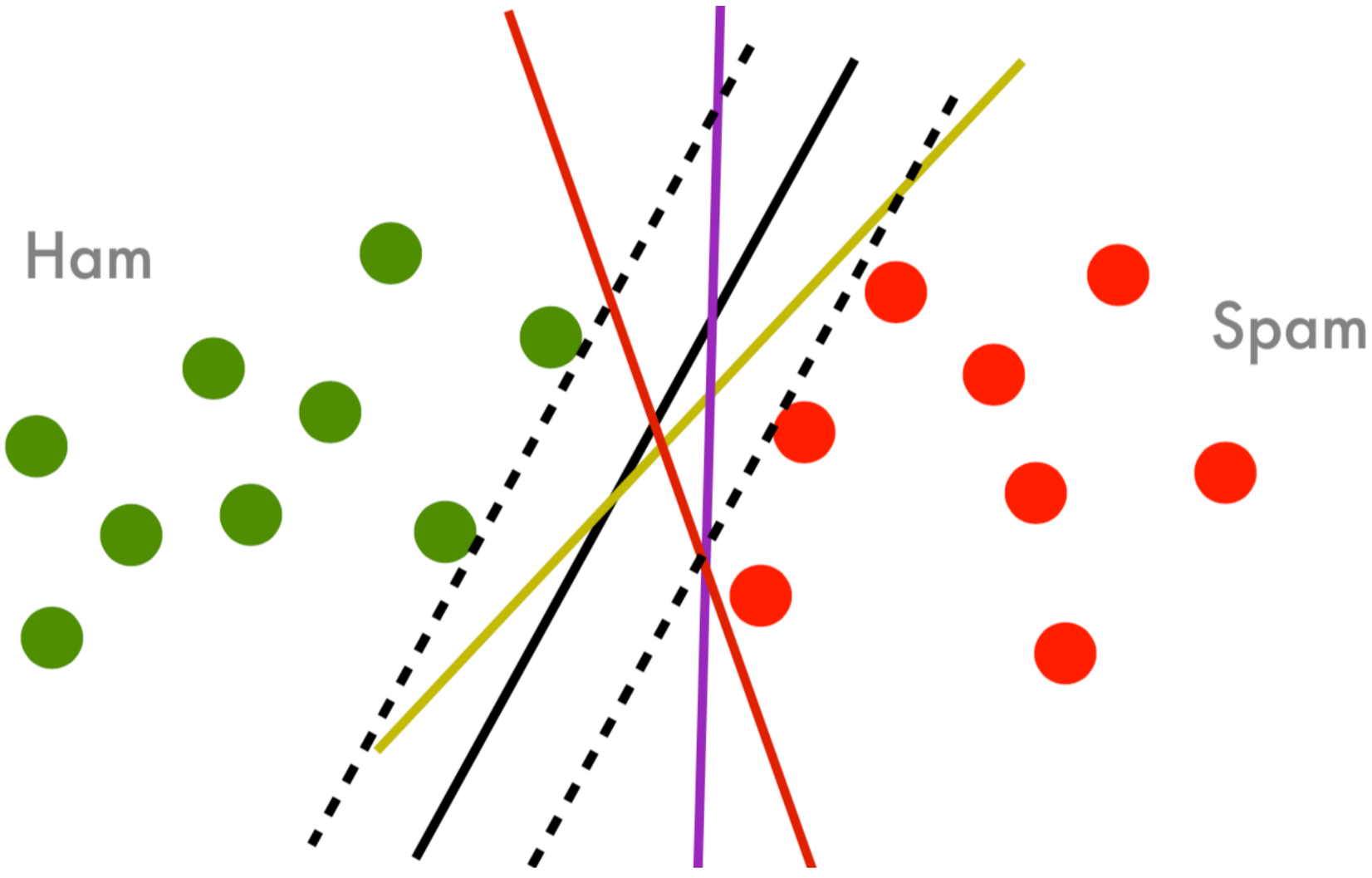
Support Vector Machines:



Key idea: find the linear classifier that maximizes the margin!

Kernel Methods: Linear SVM

Support Vector Machines:

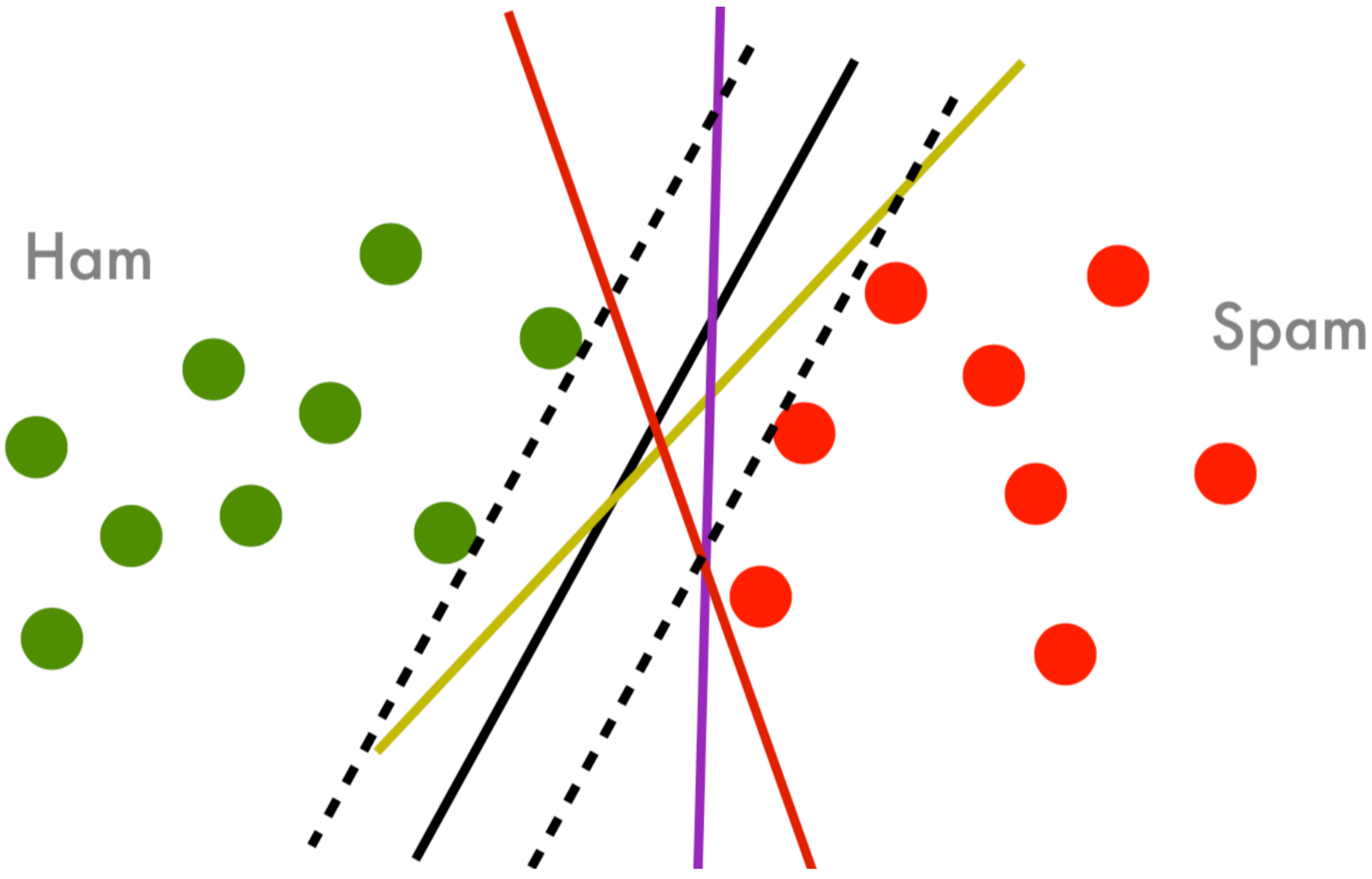


Recall one problem in our exam: how to compute the distance of a point to a given hyperplane?

Kernel Methods: Linear SVM

Support Vector Machines:

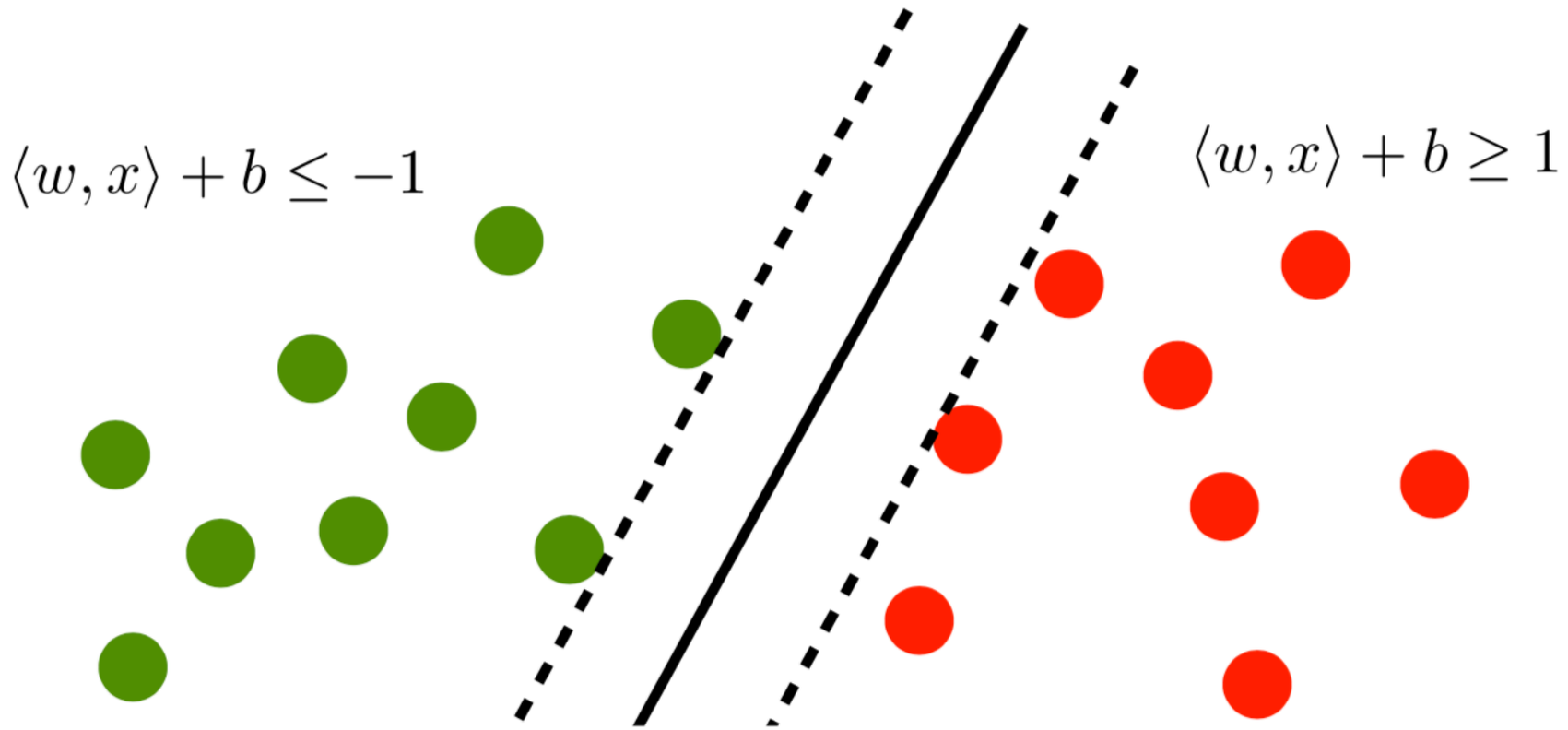
Margin: $\min_x \ |w^T x + b|$



Recall one problem in our exam: how to compute the distance of a point to a given hyperplane?

Kernel Methods: Linear SVM

Support Vector Machines:



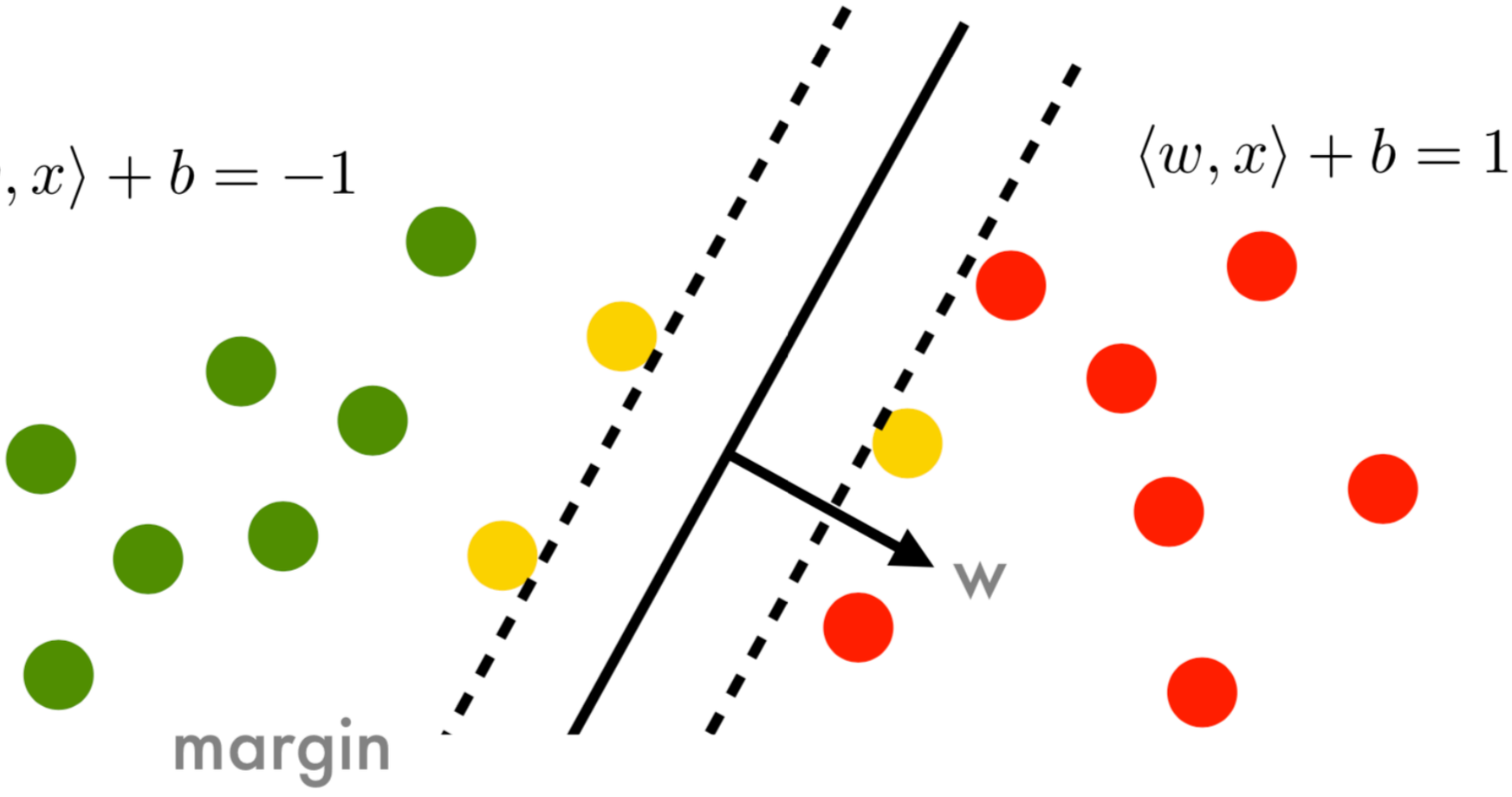
linear function

$$f(x) = \langle w, x \rangle + b$$

Think: Why can we assume that $\min_x \ |w^T x + b| = 1$ WLOG?

Kernel Methods: Linear SVM

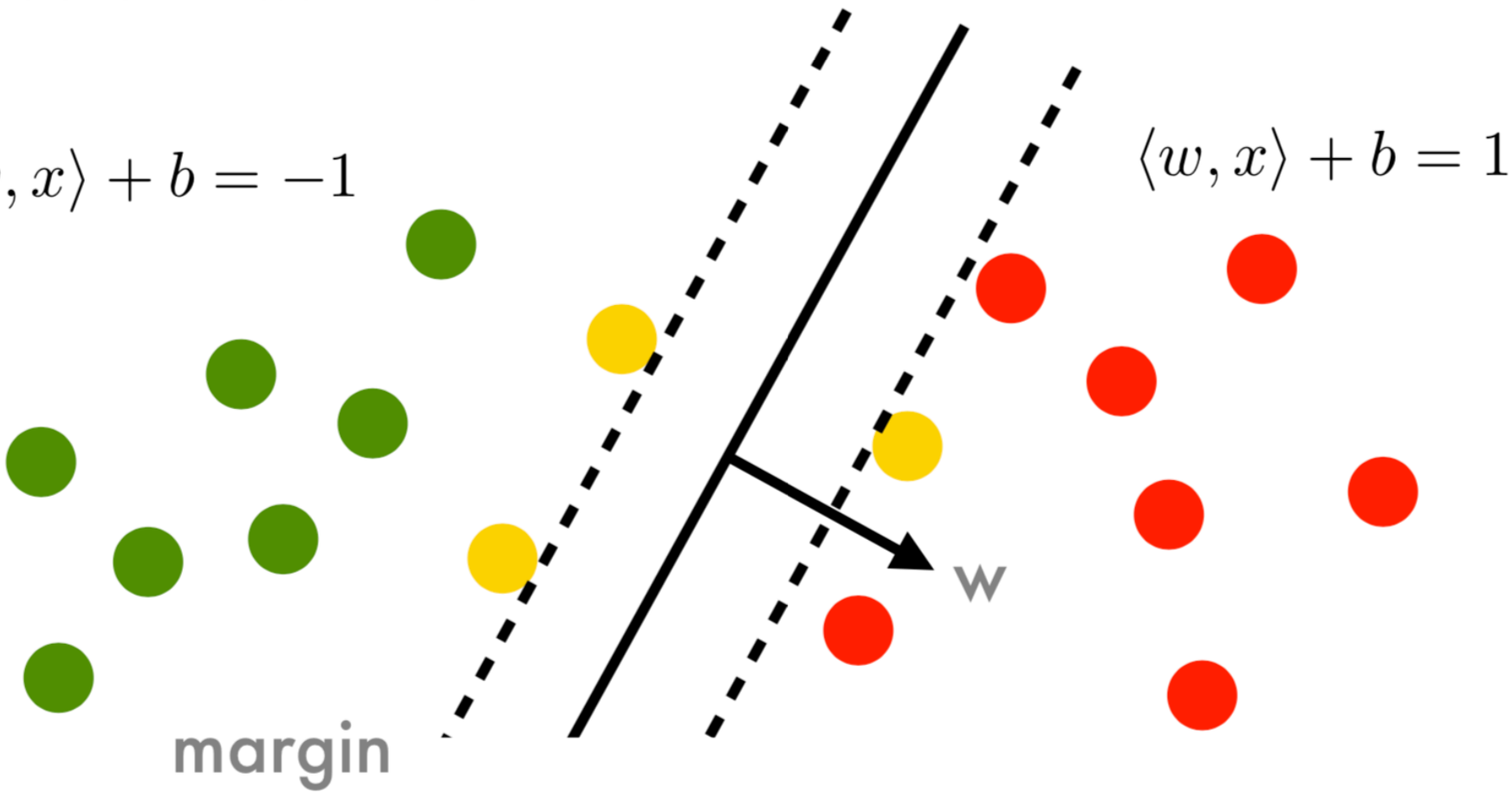
Support Vector Machines:



$$\frac{\langle x_+ - x_-, w \rangle}{2\,\|w\|} = \frac{1}{2\,\|w\|}\,[[\langle x_+, w \rangle + b] - [\langle x_-, w \rangle + b]] = \frac{1}{\|w\|}$$

Kernel Methods: Linear SVM

Support Vector Machines:



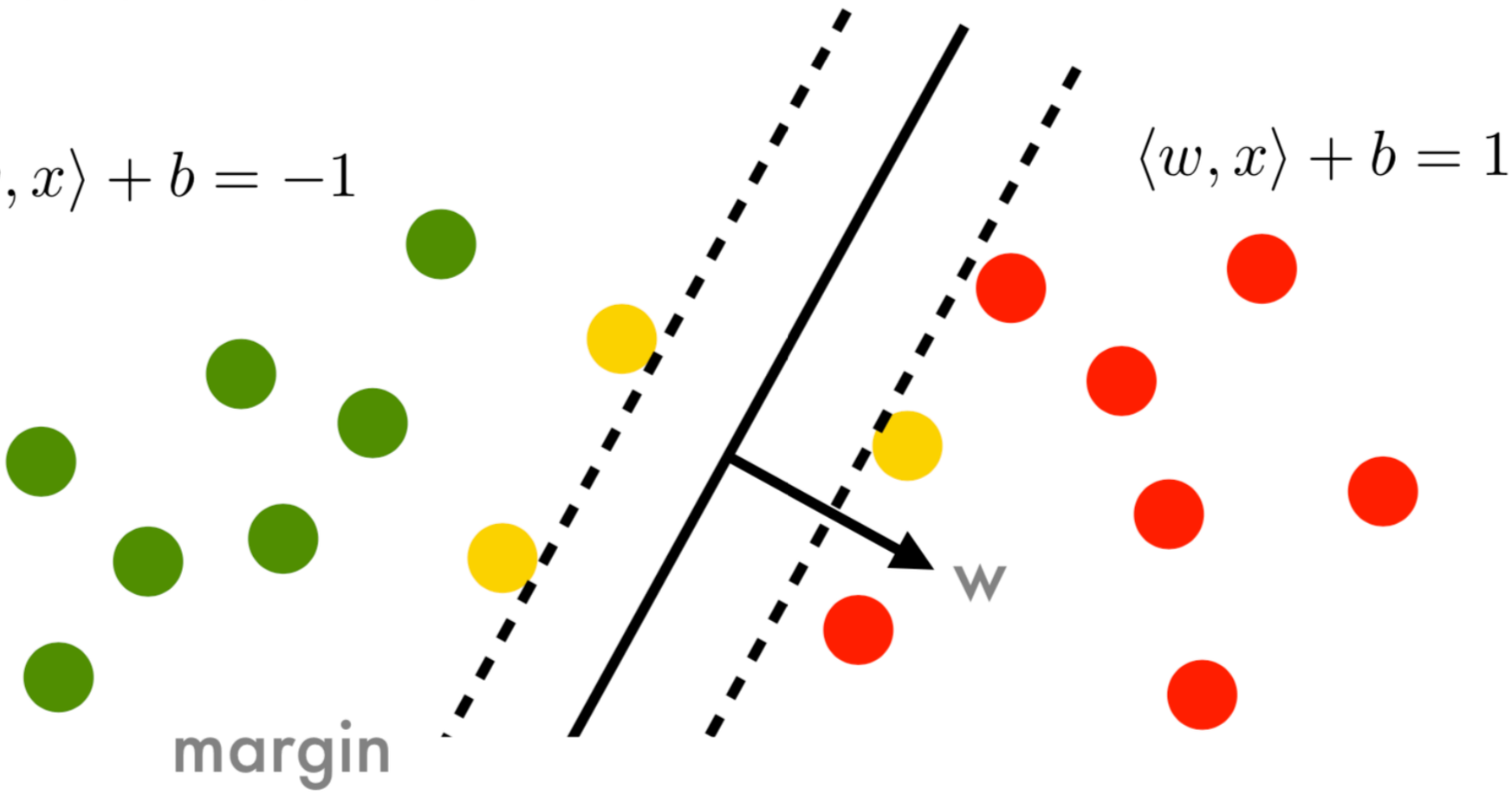
$$\frac{\langle x_+ - x_-, w \rangle}{2 \|w\|} = \frac{1}{2 \|w\|} [[\langle x_+, w \rangle + b] - [\langle x_-, w \rangle + b]] = \frac{1}{\|w\|}$$

Optimization formulation:

$$\underset{w,b}{\text{maximize}} \frac{1}{\|w\|} \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

Kernel Methods: Linear SVM

Support Vector Machines:



$$\frac{\langle x_+ - x_-, w \rangle}{2 \|w\|} = \frac{1}{2 \|w\|} [[\langle x_+, w \rangle + b] - [\langle x_-, w \rangle + b]] = \frac{1}{\|w\|}$$

Or equivalently, the following form:

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

Kernel Methods: Linear SVM

Support Vector Machines:

Constrained form:

$$\underset{w,b}{\text{minimize}}\ \frac{1}{2}\|w\|^2 \quad \text{subject to } y_i\left[\langle x_i, w\rangle + b\right] \geq 1$$

Now we consider the so-called Lagrangian function:

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i \left[y_i\left[\langle x_i, w\rangle + b\right] - 1\right], \quad \forall i, \alpha_i \geq 0$$

Claim:

- $\min_{w,b}\max_{\alpha} L(w,b,\alpha)$ gives back the original optimal solution
- Optimality in w, b must be at saddle point with alpha

This implies that the gradient of w, b at the optimality need to vanish!

Kernel Methods: Linear SVM

Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i \left[y_i \left[\langle x_i, w\rangle + b\right] - 1\right], \quad \forall i, \alpha_i \geq 0$$

Gradient:

$$\frac{\partial L(w, b, \alpha)}{\partial w} = w - \sum_i \alpha_i y_i x_i = 0 \implies w = \sum_i \alpha_i y_i x_i$$

$$\frac{\partial L(w, b, \alpha)}{\partial b} = \sum_i \alpha_i y_i = 0$$

Plugging the above two equations into the Lagrangian yields:

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_i \alpha_i$$

$$\text{subject to} \quad \sum_i \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0$$

Looks really complicated?

Kernel Methods: Linear SVM

Remember our old friend: use matrix for compact representation

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_i \alpha_i$$

$$\text{subject to} \quad \sum_i \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0$$

Define $x_i' = y_i x_i$, then:

$$\max_{\alpha} \quad -\frac{1}{2}\alpha^T X^T X \alpha + \alpha^T 1_n$$

$$\text{subject to} \quad \alpha^T y = 0 \text{ and } \alpha \geq 0$$

where X is the data matrix of x'

The above optimization problem is called Quadratic Programming (QP)

Verify: This is a convex optimization problem

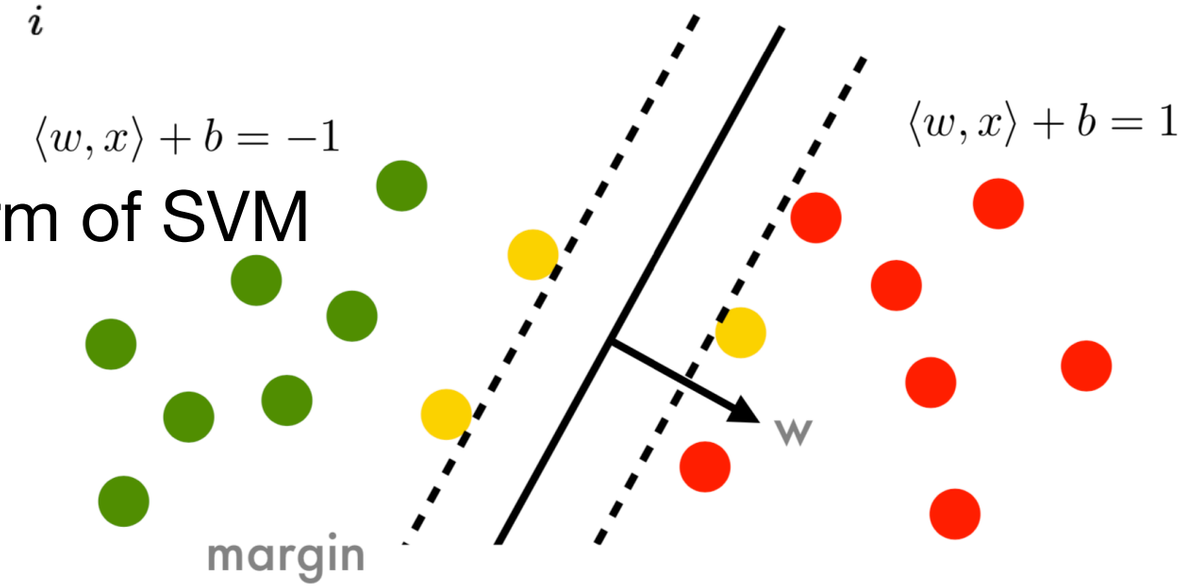
Kernel Methods: Linear SVM

Solving the QP gives us $\alpha \geq 0$

$$\max_{\alpha} \quad -\frac{1}{2}\alpha^T X^T X \alpha + \alpha^T 1_n$$
$$\text{subject to} \quad \alpha^T y = 0 \text{ and } \alpha \geq 0$$

- Note that only input vectors with $\alpha_i > 0$ matters, and they are called **support vectors**
- This can also be seen from $w = \sum_i \alpha_i y_i x_i$ i.e., only support vectors contribute to w

This QP is also known as the dual form of SVM



$$\frac{\langle x_+ - x_-, w \rangle}{2\|w\|} = \frac{1}{2\|w\|}[[\langle x_+, w \rangle + b] - [\langle x_-, w \rangle + b]] = \frac{1}{\|w\|}$$

Kernel Methods: Linear SVM

OK, this is a long journal, let have a summary before we proceed:

Hard-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j}$$

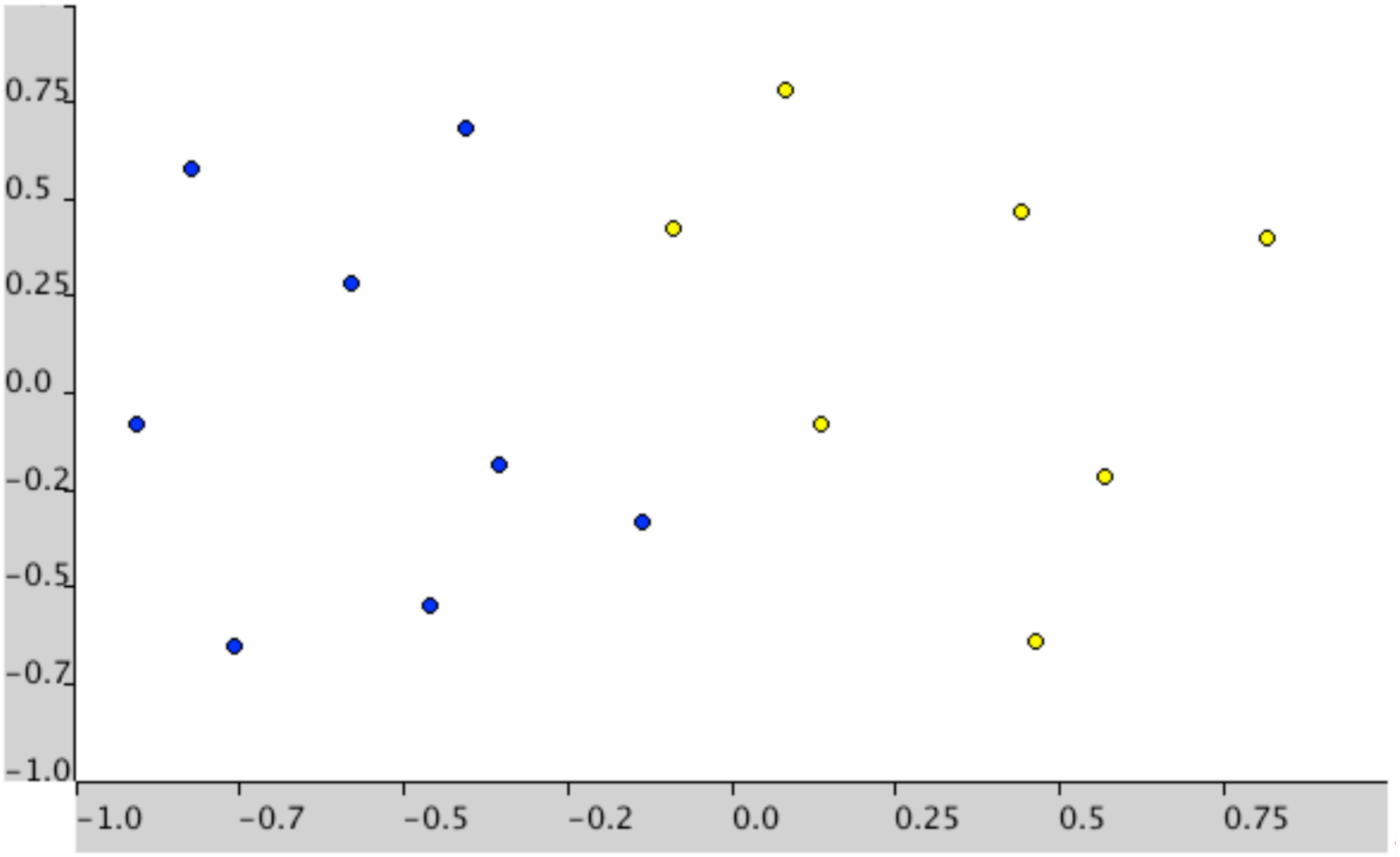
$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

- Instead of minimizing the primal, we can maximize the dual problem
- For the SVM, these two problems give the same answer (i.e. the minimum of one is the maximum of the other)

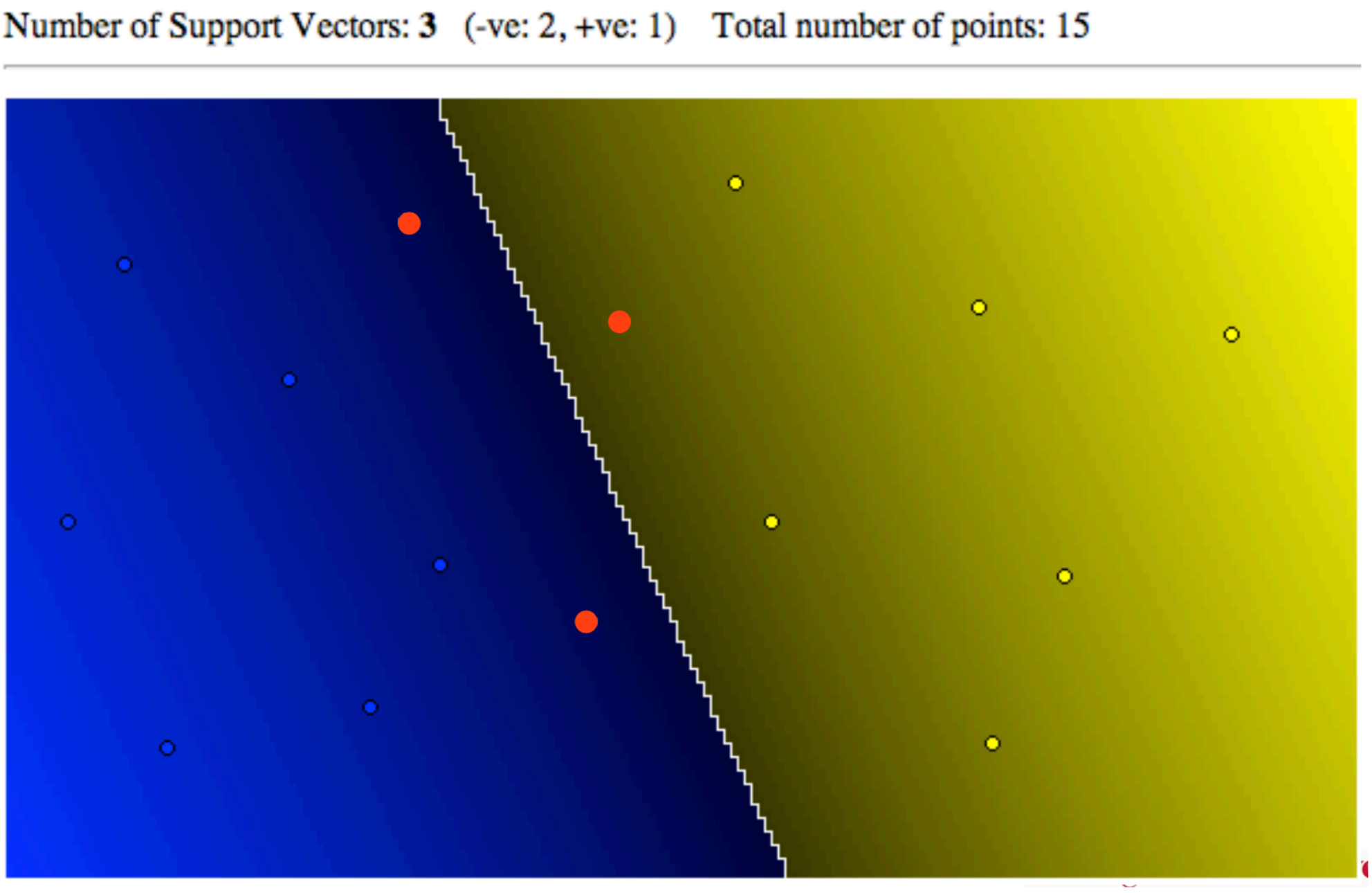
Kernel Methods: Linear SVM

Example:



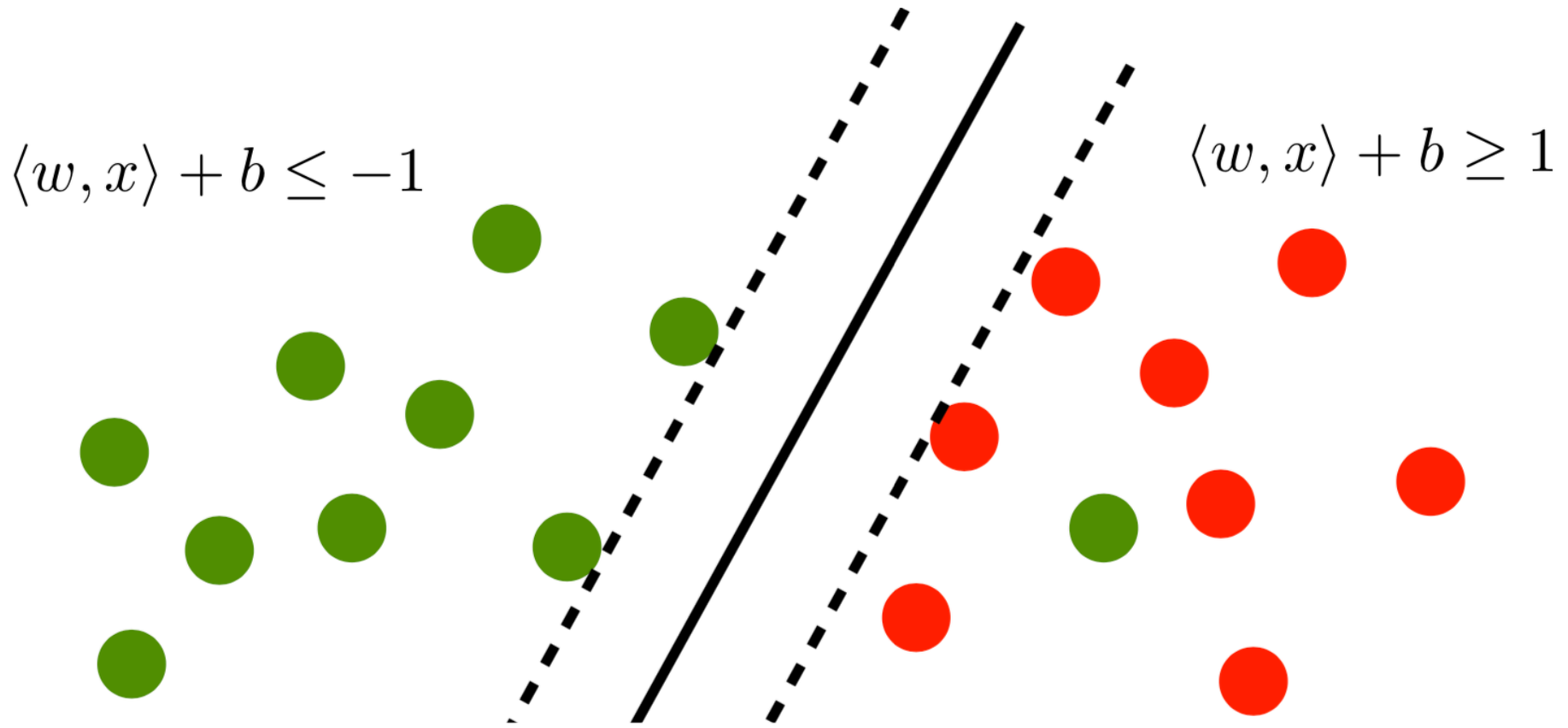
Kernel Methods: Linear SVM

Example:



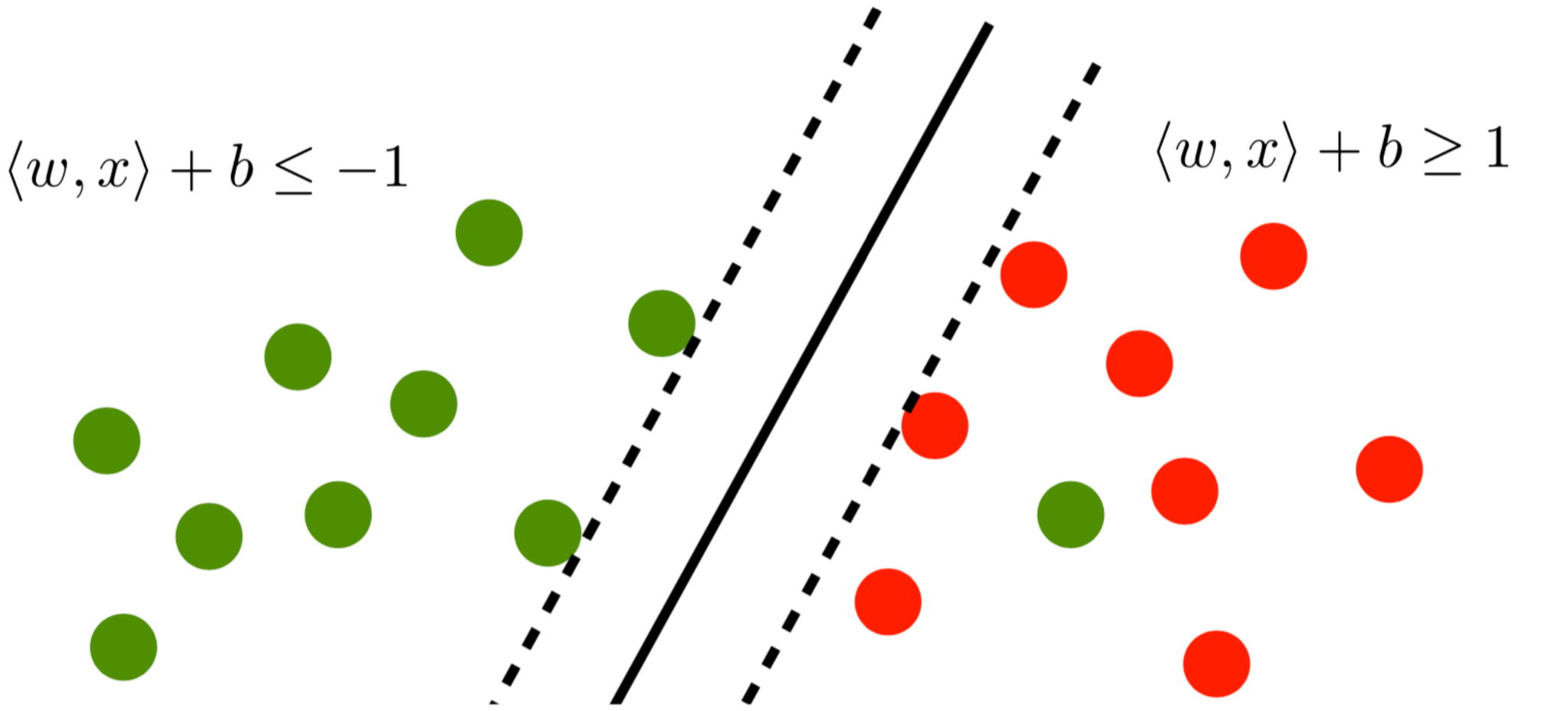
Kernel Methods: Linear SVM

The previous formulation is known as the hard-margin SVM, since we assume that data are linearly separable. However, this is hardly the case in practice



Kernel Methods: Linear SVM

On the other hand, finding the optimal hyperplane in NP-hard:

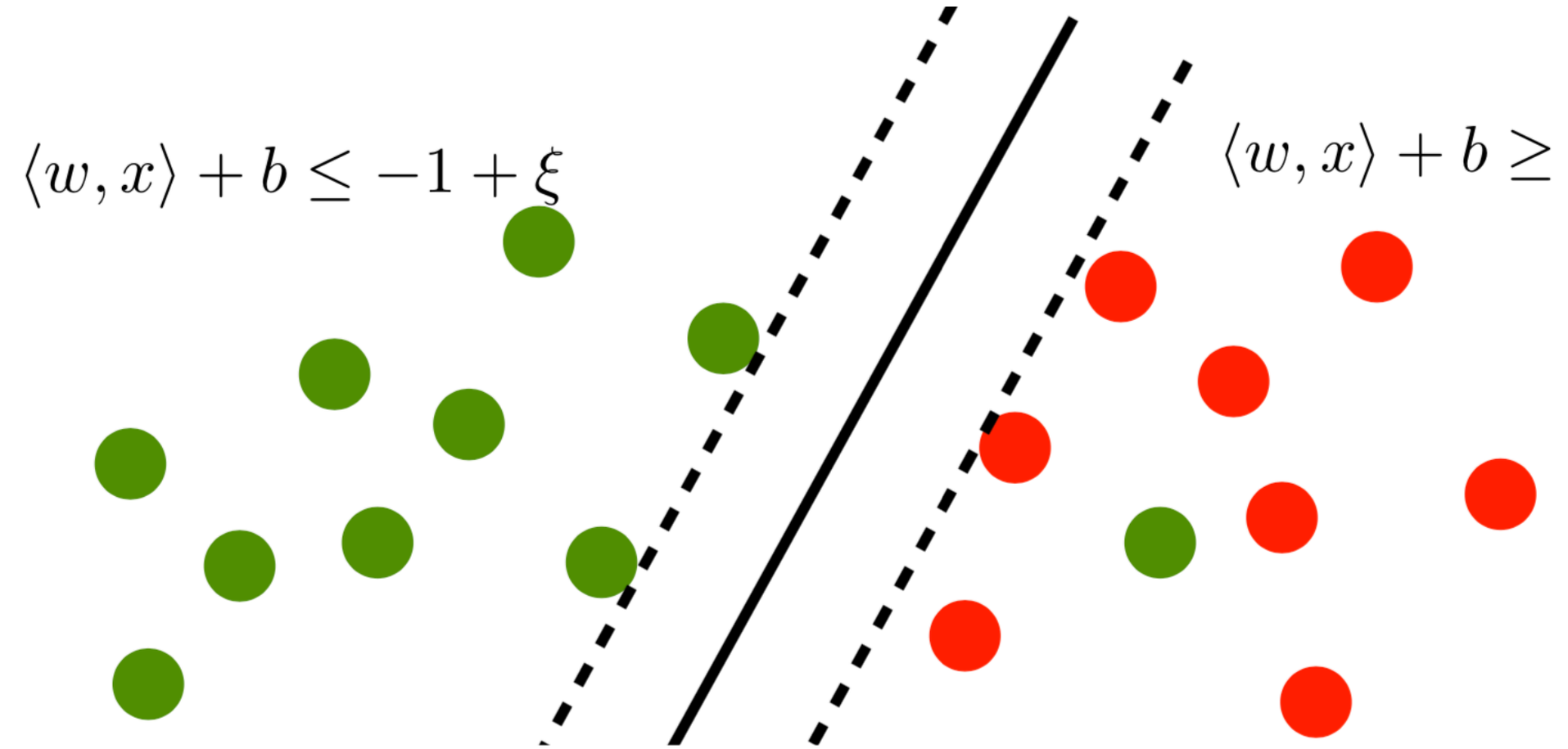


Theorem (Minsky & Papert)
Finding the minimum error separating hyperplane is NP hard

Kernel Methods: Linear SVM

Extension of hard margin linear SVM: soft margin linear SVM

Important Idea: adding slack variables to convex problems!



Convex optimization problem

Kernel Methods: Linear SVM

Soft margin Linear SVM (primal form):

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

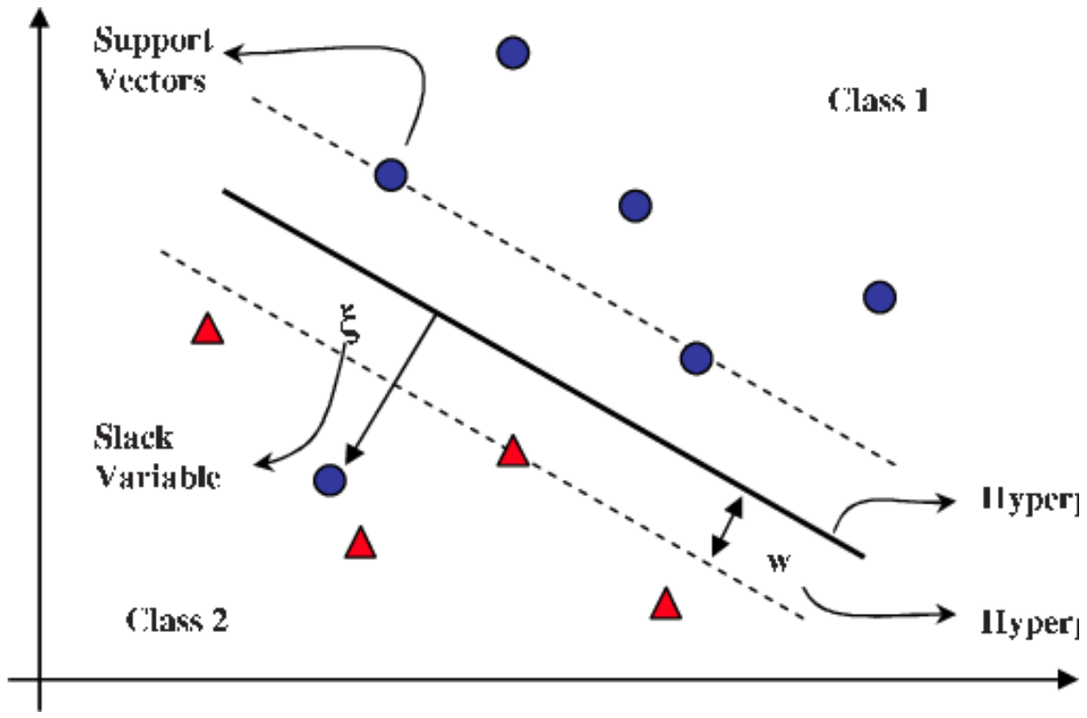
$$\text{subject to } y_i\left[\langle w, x_i\rangle + b\right] \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

Problem is always feasible. $w = 0$ and $b = 0$ and $\xi_i = 1$

Think: what does it mean if

- $\xi_i = 0$
- $0 < \xi_i \leq 1$
- $\xi_i > 1$

Think: what's the meaning of C?



Kernel Methods: Linear SVM

Think: Apply the previous Lagrangian method, show the dual form of the soft margin linear SVM has the following form:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\alpha^T X^T X \alpha + \alpha^T 1_n \\ \text{subject to} \quad & \alpha^T y = 0 \text{ and } 0 \leq \alpha \leq C \end{aligned}$$

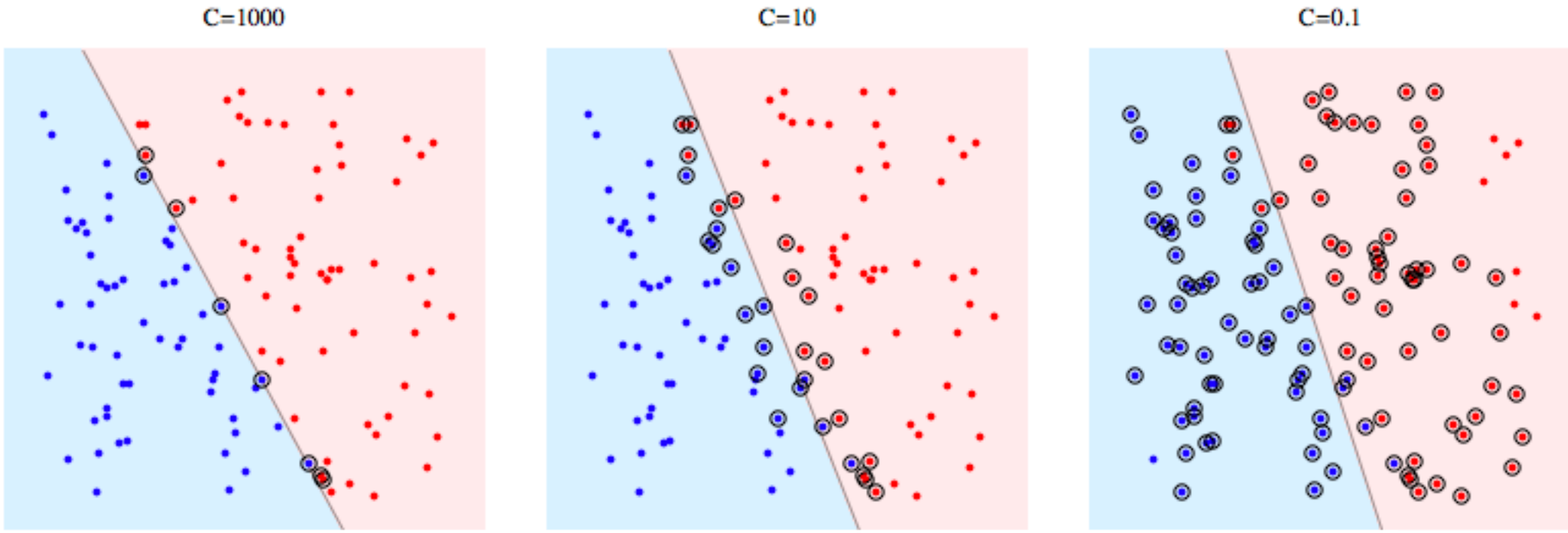
(Hint) You can go through the following steps:

1. Write down the Lagrangian by first converting constrained form to unconstrained form
2. Use the saddle point argument to show that for optimality, gradient w.r.t. w and b must vanish
3. Plugging into the above gradient solution of w and b, then simplify

Good luck :)

Kernel Methods: Linear SVM

In practice, the effect of C:



$$\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

$$\text{subject to } y_i\left[\langle w, x_i\rangle + b\right] \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

Larger C: less violation of linearity allowed

Smaller C: more violation of linearity allowed

Kernel Methods: Linear SVM

Let's take a closer look at the primal form of SVM:

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

$$\text{subject to } y_i\left[\langle w, x_i\rangle + b\right] \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

This is the constrained form of a convex optimization problem, what's its penalty form?

$$\min_{w,b} \quad \frac{1}{2}||w||^2 + C\sum_i \max\{1 - y_i(\langle w, x_i\rangle + b), 0\}$$

Kernel Methods: Linear SVM

Let's take a closer look at the primal form of SVM:

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

$$\text{subject to } y_i\left[\langle w, x_i\rangle + b\right] \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

This is the constrained form of a convex optimization problem, what's its penalty form?

$$\min_{w,b} \quad \frac{1}{2}||w||^2 + C\sum_i \max\{1 - y_i(\langle w, x_i\rangle + b), 0\}$$

Now let's use the same trick to incorporate the intercept term (b) into our vector representation by adding an additional 1 into our feature representation:

$$\min_{w,b} \quad \frac{1}{2}||w||^2 + C\sum_i (1 - y_i x_i^T w)_+$$

Hinge loss

Kernel Methods: Linear SVM

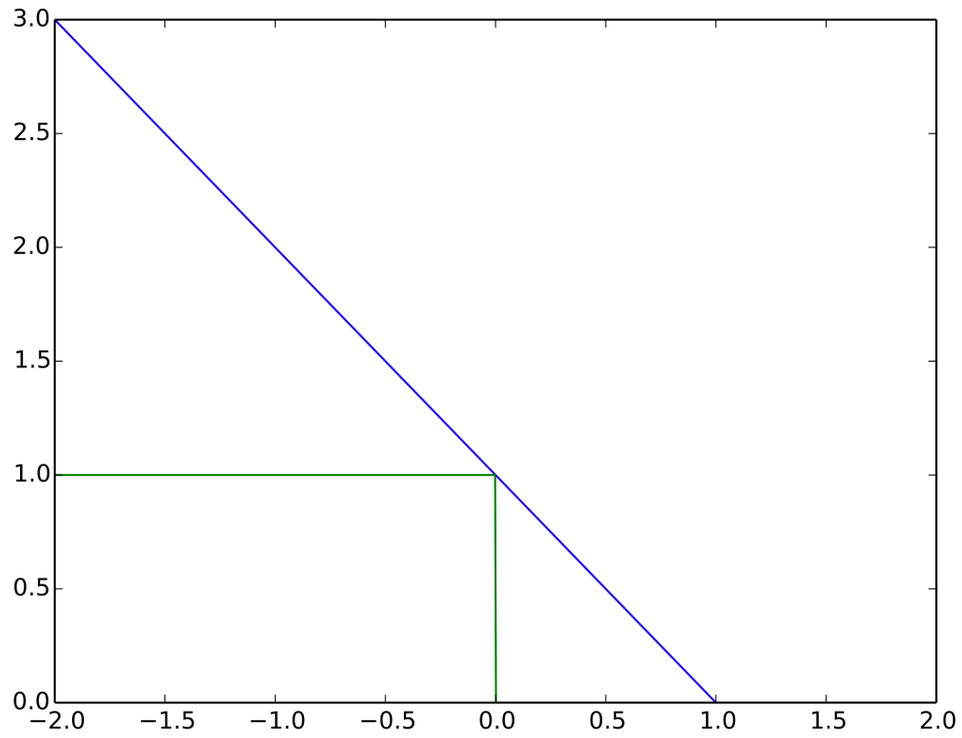
Let's take a closer look at the primal form of SVM:

$$\min_{w,b} \quad \frac{1}{2}||w||^2 + C\sum_i (1 - y_i x_i^T w)_+$$

Hinge loss

Observation:

- SVM = Hinge loss + L2 regularization

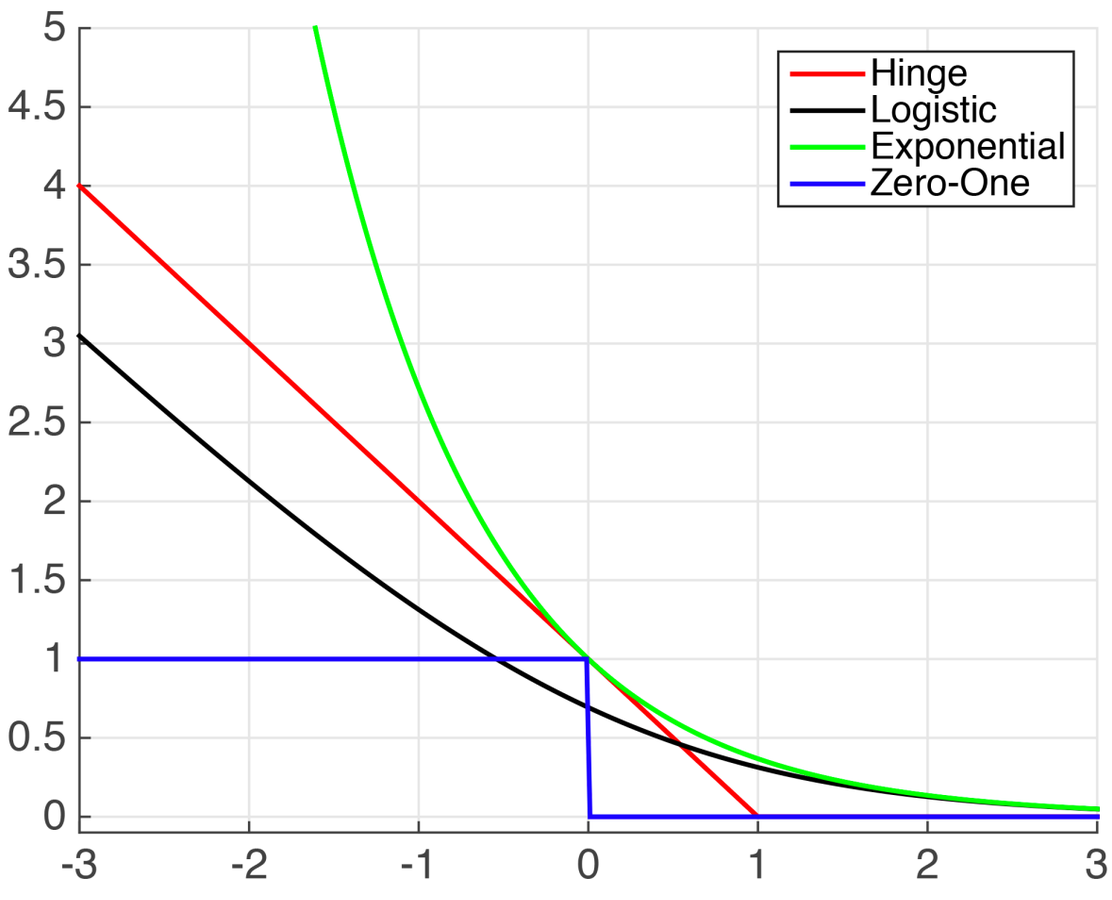


Kernel Methods: Linear SVM

In general, for linear models, we have the following form of optimization:

$$\min_{w,b} \quad \frac{\lambda}{2}||w||_2^2 + \sum_i \ell(y_i x_i^T w)$$

Depending on the choice of different loss functions, we recover different models:



Hinge: SVM

Logistic: Logistic Regression

Exponential: AdaBoost

Zero-One: No (NP-hard)

Kernel Methods: Linear SVM

Summary of Linear SVM in primal/dual hard/soft margin scenario:

Hard-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

Soft-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\left(\sum_{i=1}^{N} e_i\right)$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - e_i, \quad \forall i = 1, \ldots, N$$

$$e_i \geq 0, \quad \forall i = 1, \ldots, N$$

Soft-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } 0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

We can also work with the dual of the soft-margin SVM

Lecture 4: Kernel Methods and Graphical Models

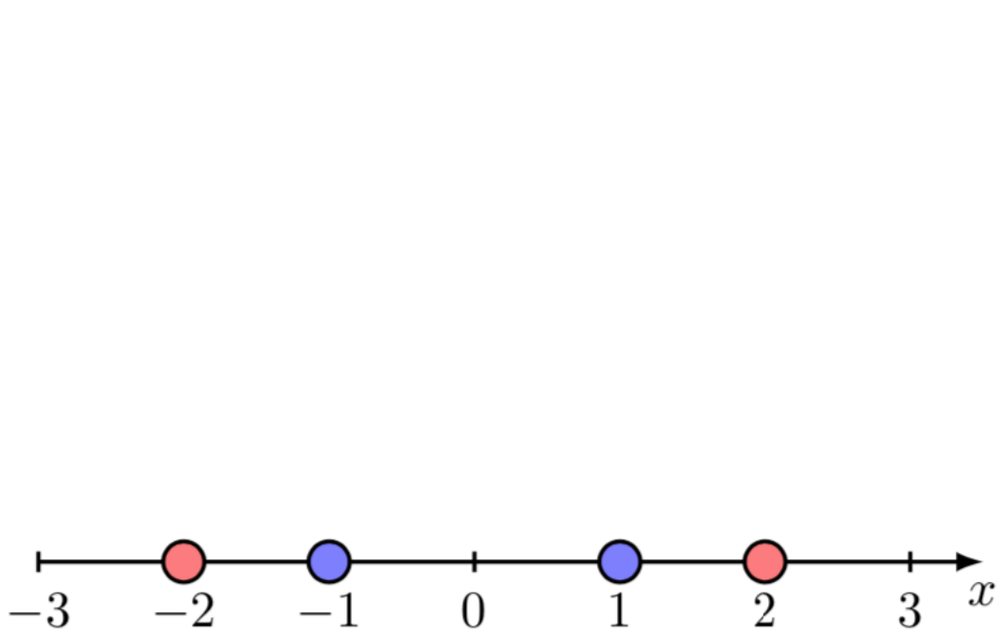
Overview:

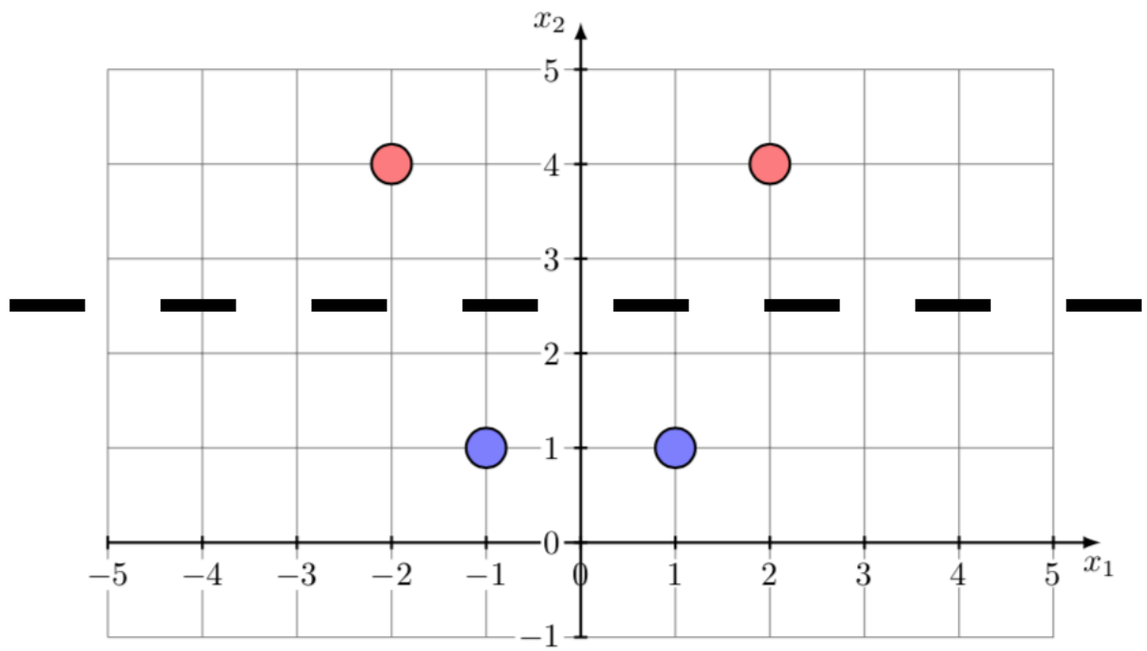
- Kernel Methods
  - Linear Support Vector Machines
  - Nonlinear Support Vector Machines

Kernel Methods: Nonlinear SVM

How to generalize linear SVM to nonlinear case?

Key idea: In order to obtain nonlinear SVM, it suffices if we define nonlinear feature map $\phi : \mathbb{R}^d \to \mathcal{H}$, where $\mathcal{H}$ is called a Hilbert space



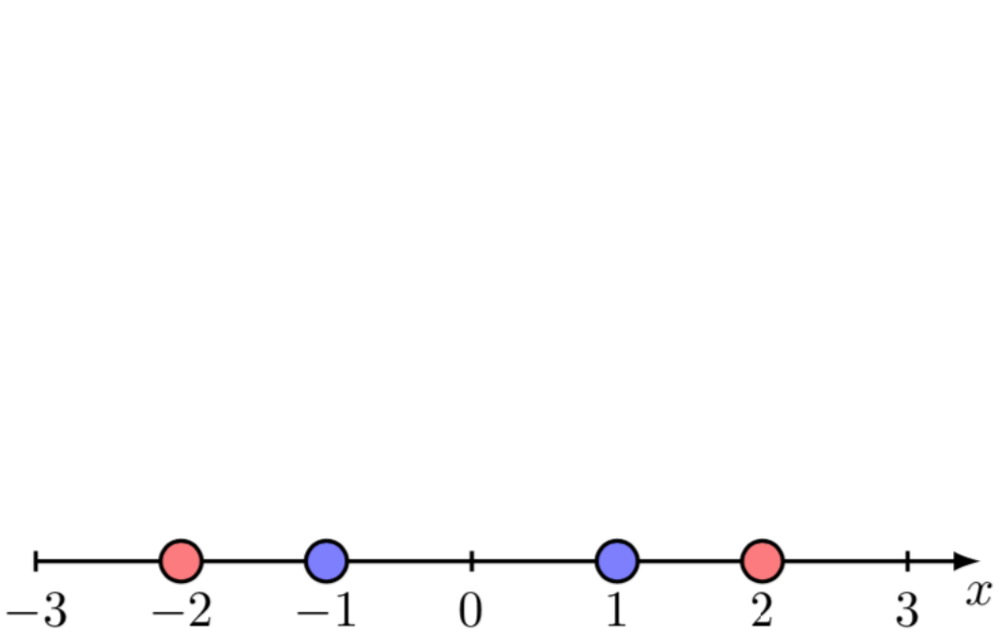


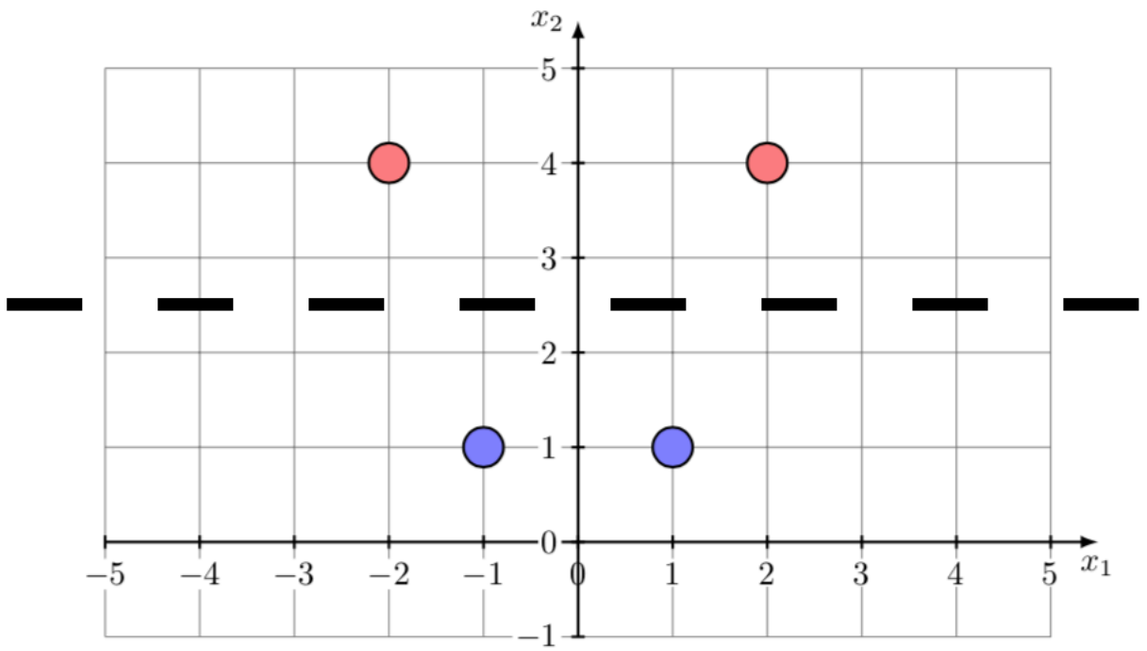
Feature map: $\phi(x) = (x, x^2)$

Data become linearly separable after the feature map in the new space!

Kernel Methods: Nonlinear SVM

Key idea: In order to obtain nonlinear SVM, it suffices if we define nonlinear feature map $\phi : \mathbb{R}^d \to \mathcal{H}$, where $\mathcal{H}$ is called a Hilbert space





However, the dimension becomes larger. Curse of dimensionality?

Not necessarily if we can compute the inner product in the new space efficiently! And if so, we obtain nonlinear extension of SVM for free!

Kernel Methods: Nonlinear SVM

Recall the dual optimization problem of soft margin SVM:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\alpha^T X^T X \alpha + \alpha^T 1_n \\ \text{subject to} \quad & \alpha^T y = 0 \text{ and } 0 \leq \alpha \leq C \end{aligned}$$

The matrix $X^T X$ is called the Gram matrix, and it has the following special structure:

$$X^T X := \begin{vmatrix} \langle y_1 x_1, y_1 x_1 \rangle & \langle y_1 x_1, y_2 x_2 \rangle & \dots & \langle y_1 x_1, y_n x_n \rangle \\ \langle y_2 x_2, y_1 x_1 \rangle & \langle y_2 x_2, y_2 x_2 \rangle & \dots & \langle y_2 x_2, y_n x_n \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle y_n x_n, y_1 x_1 \rangle & \langle y_n x_n, y_2 x_2 \rangle & \dots & \langle y_n x_n, y_n x_n \rangle \end{vmatrix}$$

i.e., it's the inner product between data using the Euclidean inner product

Kernel Methods: Nonlinear SVM

Recall the dual optimization problem of soft margin SVM:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\alpha^T X^T X \alpha + \alpha^T 1_n \\ \text{subject to} \quad & \alpha^T y = 0 \text{ and } 0 \leq \alpha \leq C \end{aligned}$$

The matrix $X^T X$ is called the Gram matrix, and it has the following special structure:

$$X^T X := \begin{vmatrix} \langle y_1 x_1, y_1 x_1 \rangle & \langle y_1 x_1, y_2 x_2 \rangle & \dots & \langle y_1 x_1, y_n x_n \rangle \\ \langle y_2 x_2, y_1 x_1 \rangle & \langle y_2 x_2, y_2 x_2 \rangle & \dots & \langle y_2 x_2, y_n x_n \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle y_n x_n, y_1 x_1 \rangle & \langle y_n x_n, y_2 x_2 \rangle & \dots & \langle y_n x_n, y_n x_n \rangle \end{vmatrix}$$

i.e., it's the inner product between data using the Euclidean inner product

Key idea: In order to obtain nonlinear SVM, it suffices if we define nonlinear feature map $\phi : \mathbb{R}^d \to \mathcal{H}$, where $\mathcal{H}$ is called a Hilbert space

Kernel Methods: Nonlinear SVM

From now on we only focus on the soft-margin SVM dual form:

Hard-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

Soft-margin SVM (Primal)

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\left(\sum_{i=1}^{N} e_i\right)$$

$$\text{s.t. } y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - e_i, \quad \forall i = 1, \ldots, N$$

$$e_i \geq 0, \quad \forall i = 1, \ldots, N$$

Soft-margin SVM (Lagrangian Dual)

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } 0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

We can also work with the dual of the soft-margin SVM

Kernel Methods: Nonlinear SVM

First, let's consider quadratic features:

Quadratic Features in $\mathbb{R}^2$

$$\Phi(x) := \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)$$

Dot Product

$$\begin{aligned}\langle \Phi(x), \Phi(x') \rangle &= \left\langle \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right), \left({x'_1}^2, \sqrt{2}x'_1x'_2, {x'_2}^2\right) \right\rangle \\ &= \langle x, x' \rangle^2.\end{aligned}$$

Insight

Trick works for any polynomials of order d via $\langle x, x' \rangle^d$.

In this example we manage to not compute the feature $\phi(x)$ explicitly but still being able to compute their inner product implicitly via $\langle x, x' \rangle^2$

Kernel Methods: Nonlinear SVM

Kernel:

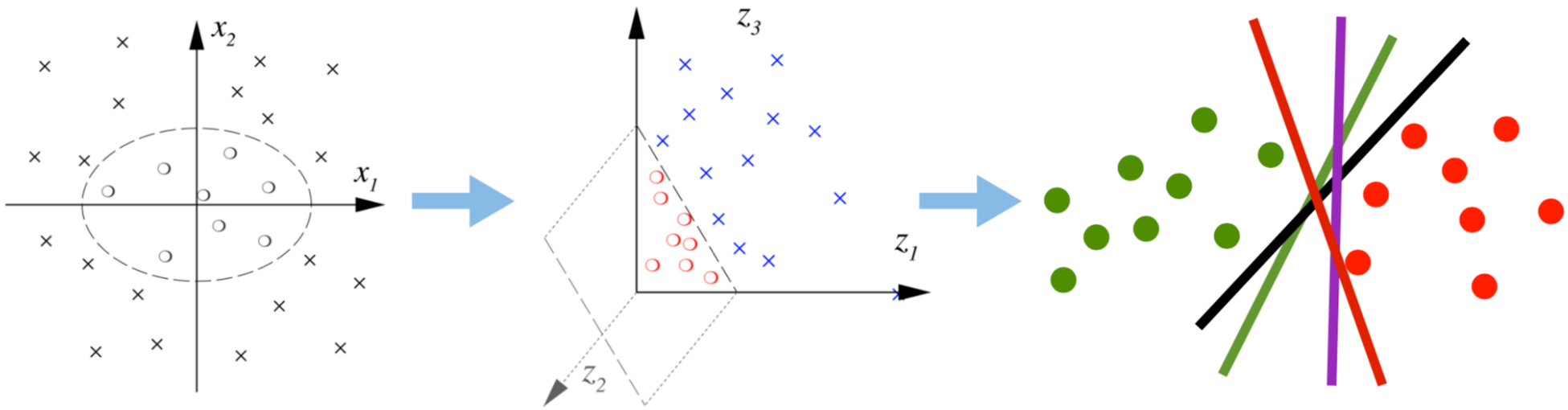
A kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric function in its arguments for which the following property holds:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$$

for some feature map $\phi : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is a (infinite dimensional) Hilbert space.

Implications:

- If the kernel function $k(\cdot, \cdot)$ is easy and efficient to compute, then we get an extension of linear SVM to nonlinear SVM for free



Kernel Methods: Nonlinear SVM

Kernel trick in dual form of SVM:

$$K := \begin{vmatrix} \langle y_1\phi(x_1), y_1\phi(x_1)\rangle & \langle y_1\phi(x_1), y_2\phi(x_2)\rangle & \dots & \langle y_1\phi(x_1), y_n\phi(x_n)\rangle \\ \langle y_2\phi(x_2), y_1\phi(x_1)\rangle & \langle y_2\phi(x_2), y_2\phi(x_2)\rangle & \dots & \langle y_2\phi(x_2), y_n\phi(x_n)\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle y_n\phi(x_n), y_1\phi(x_1)\rangle & \langle y_n\phi(x_n), y_2\phi(x_2)\rangle & \dots & \langle y_n\phi(x_n), y_n\phi(x_n)\rangle \end{vmatrix}$$

$$= \begin{vmatrix} y_1^2K(x_1,x_1) & y_1y_2K(x_1,x_2) & \dots & y_1y_nK(x_1,x_n) \\ y_2y_1K(x_2,x_1) & y_2^2K(x_2,x_2) & \dots & y_2y_nK(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ y_ny_1K(x_n,x_1) & y_ny_2K(x_n,x_2) & \dots & y_n^2K(x_n,x_n) \end{vmatrix}$$

By Mercer's theorem, it is guaranteed that for a kernel function, $K \succeq 0$ for any input sequence $\{x_i\}_{i=1}^n$

Kernel Methods: Nonlinear SVM

Kernel SVM (Nonlinear SVM) in dual form:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\alpha^T K \alpha + \alpha^T 1_n \\ \text{subject to} \quad & \alpha^T y = 0 \text{ and } 0 \leq \alpha \leq C \end{aligned}$$

- Again, this is still a QP, and can be solved as before
- The only thing changed here is that we replace the original $X^T X$ with a kernel matrix K

As a special case, if we choose our kernel function to be:

$$k(x, x') = x^T x' = \langle x, x' \rangle$$

then we recover the original linear SVM

Note: This is known as the kernel trick, a general technique to boost linear models to nonlinear extensions

Kernel Methods: Nonlinear SVM

Common kernel functions:

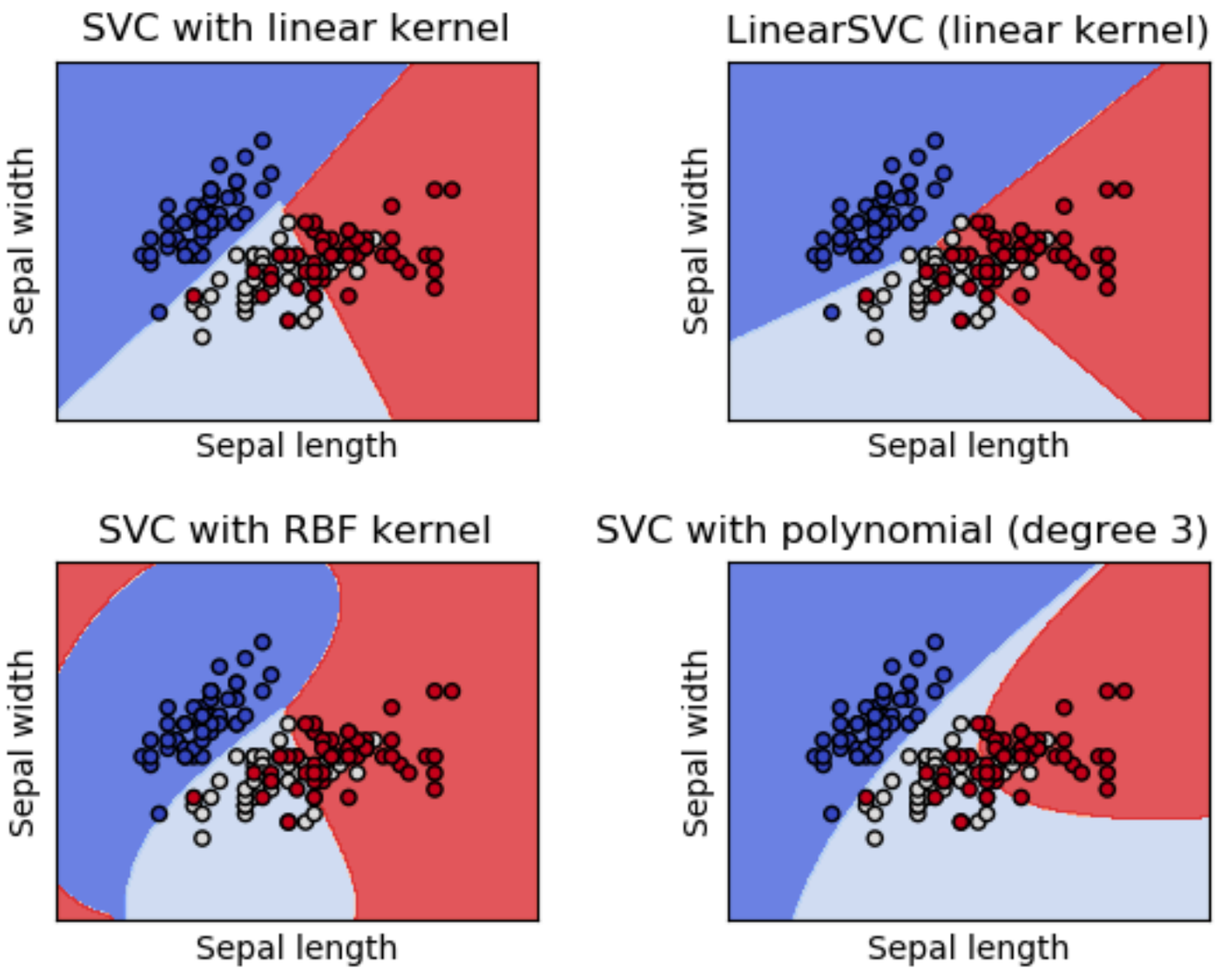
Linear	$\langle x, x' \rangle$
Laplacian RBF	$\exp\left(-\lambda \|x - x'\|\right)$
Gaussian RBF	$\exp\left(-\lambda \|x - x'\|^2\right)$
Polynomial	$\left(\langle x, x' \rangle + c\right)^d, c \geq 0, \ d \in \mathbb{N}$
B-Spline	$B_{2n+1}(x - x')$
Cond. Expectation	$\mathbf{E}_c[p(x\|c)p(x'\|c)]$

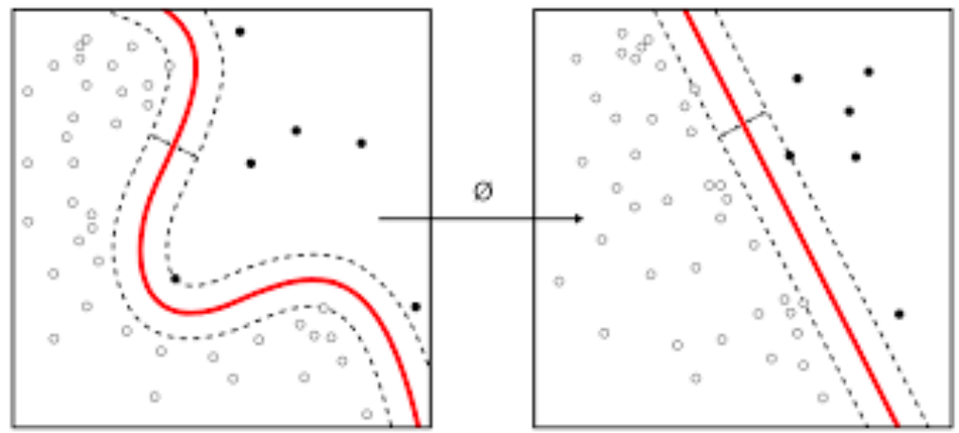
The most frequently used one

Think: how to show that the polynomial kernel is indeed a kernel?

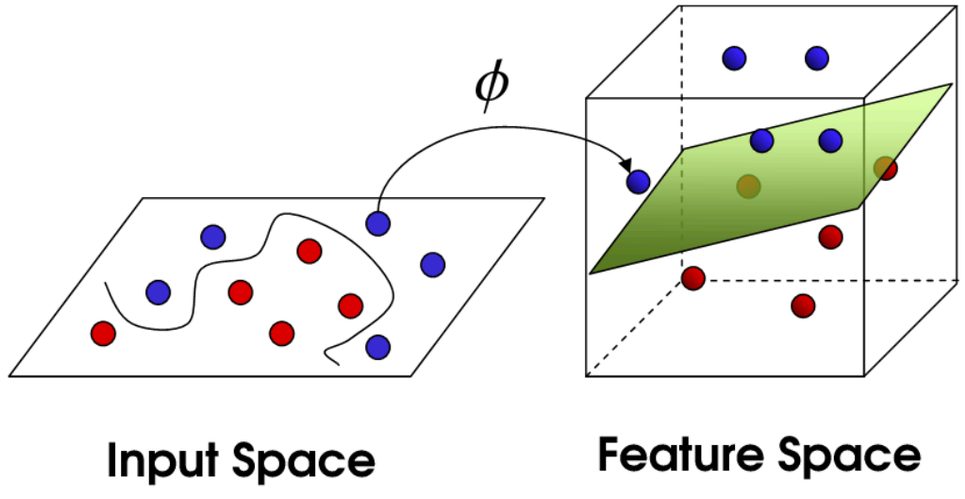
Kernel Methods: Nonlinear SVM

Kernel SVM visualization:





Iris dataset, 2 features for visualization purpose



Kernel Methods: Nonlinear SVM

SVM: multiclass classification extension

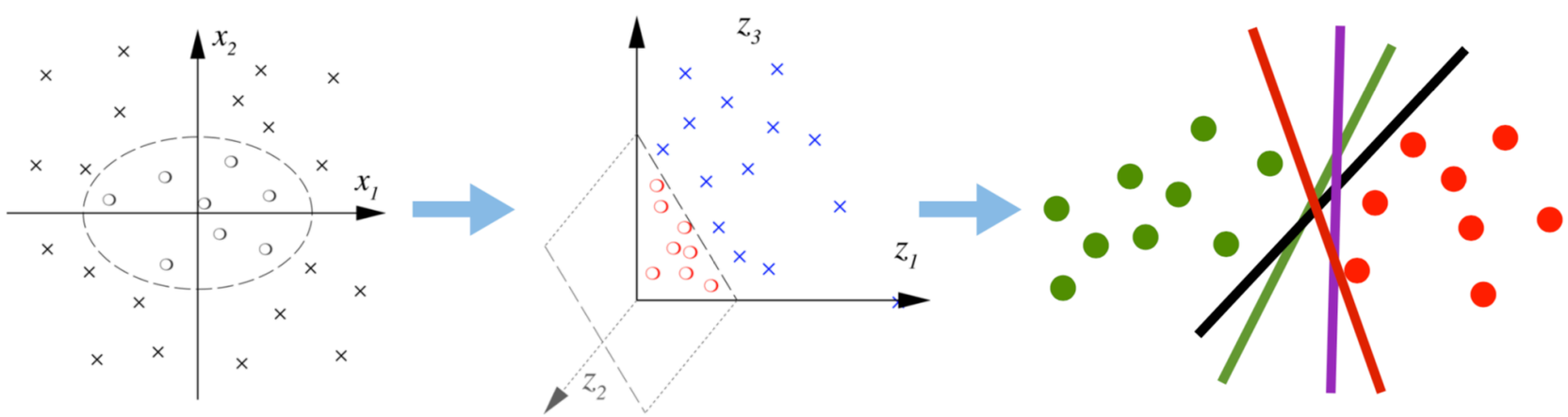
- One-vs-Rest:
  - build C classifiers
  - train the i-th classifier to predict whether an instance has label i or not
  - predict the class with the largest score

- One-vs-One:
  - build (C choose 2) binary classifiers
  - train one classifier to distinguish between each pair of labels
  - predict the class with the most votes from any given classifier

Kernel Methods: Nonlinear SVM

Kernel methods are too hard…What should I know and remember?

- Understand the principle of learning max-margin decision boundary
- Understand the connection between constrained and unconstrained convex optimization
- Understand the hard-margin linear SVM derivation
- Follow the proof for Lagrangian dual form of SVM
- Describe the mathematical properties of support vectors and being able to provide an intuitive explanation for them
- Understand the pipeline of kernel SVM



Lecture 4: Homework

Good news for you:

- You don’t have any math HW in this section since the content in this section might be too hard/abstract to understand

But, to prepare for our incoming Deep Learning section, you’re asked to implement a multi-class Logistic Regression for digit classification using PyTorch on MNIST