SVMC

An introduction to Support Vector Machines Classification

6.783, Biomedical Decision Support

Lorenzo Rosasco
(lrosasco@mit.edu)
Department of Brain and Cognitive Science
MIT

A typical problem

- We have a cohort of patients from two groups- say A and B.
- We wish to devise a classification rule to distinguish patients of one group from patients of the other group.

Learning and Generalization

Goal: classify correctly new patients

Plan

1. Linear SVM
2. Non Linear SVM: Kernels
3. Tuning SVM
4. Beyond SVM: Regularization Networks

Learning from Data

To make predictions we need informations about the patients

patient 1: $x = (x^1, \ldots, x^n)$

patient 2 : $x = (x^1, \ldots, x^n)$

....

patient ℓ : $x = (x^1, \ldots, x^n)$

Linear model

Patients of class A are labeled y=1

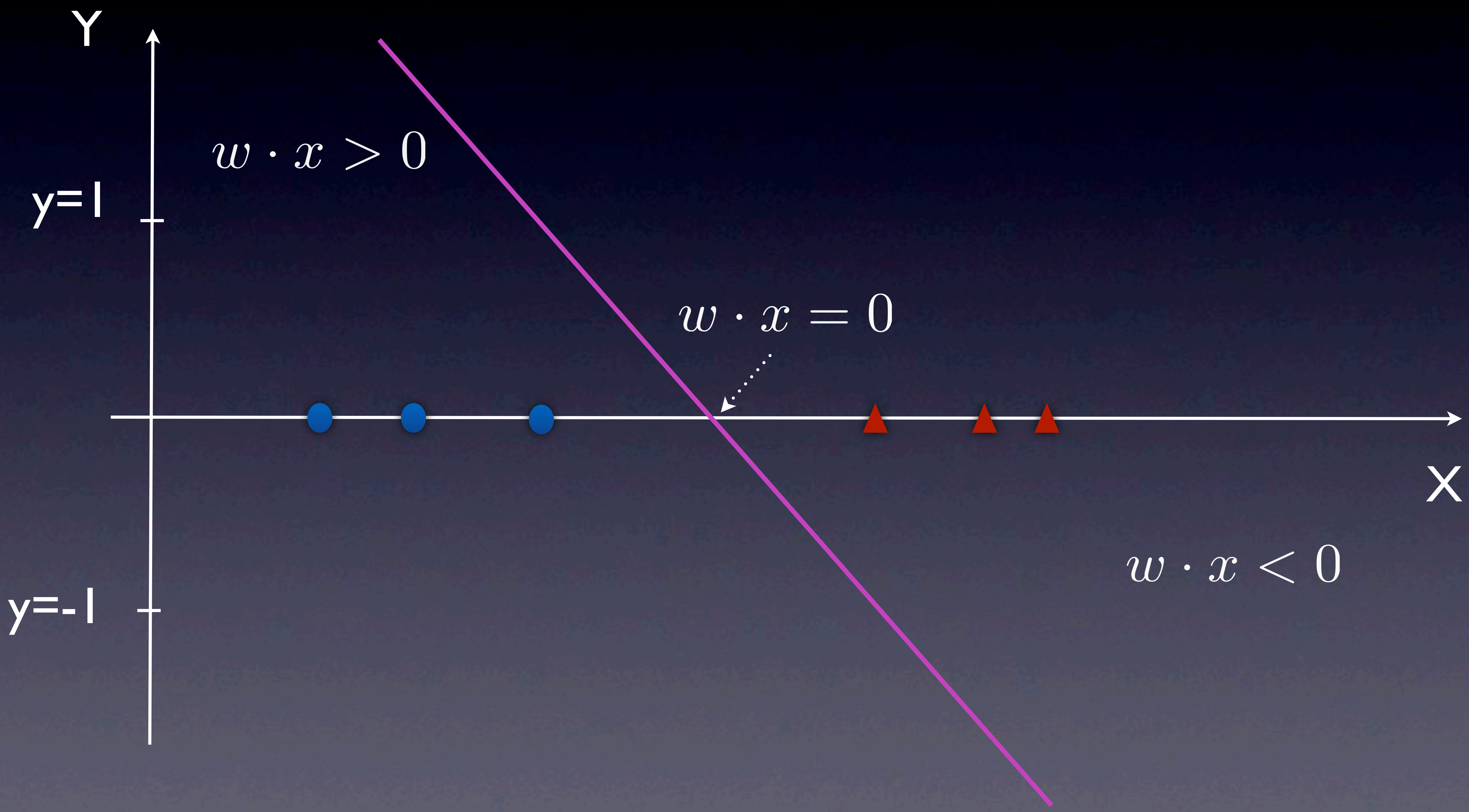
Patients of class B are labeled y=-1

Linear model

$$w \cdot x = \sum_{j=1}^{n} w_j x^j$$

classification rule $\operatorname{sign}(w \cdot x)$

1D Case

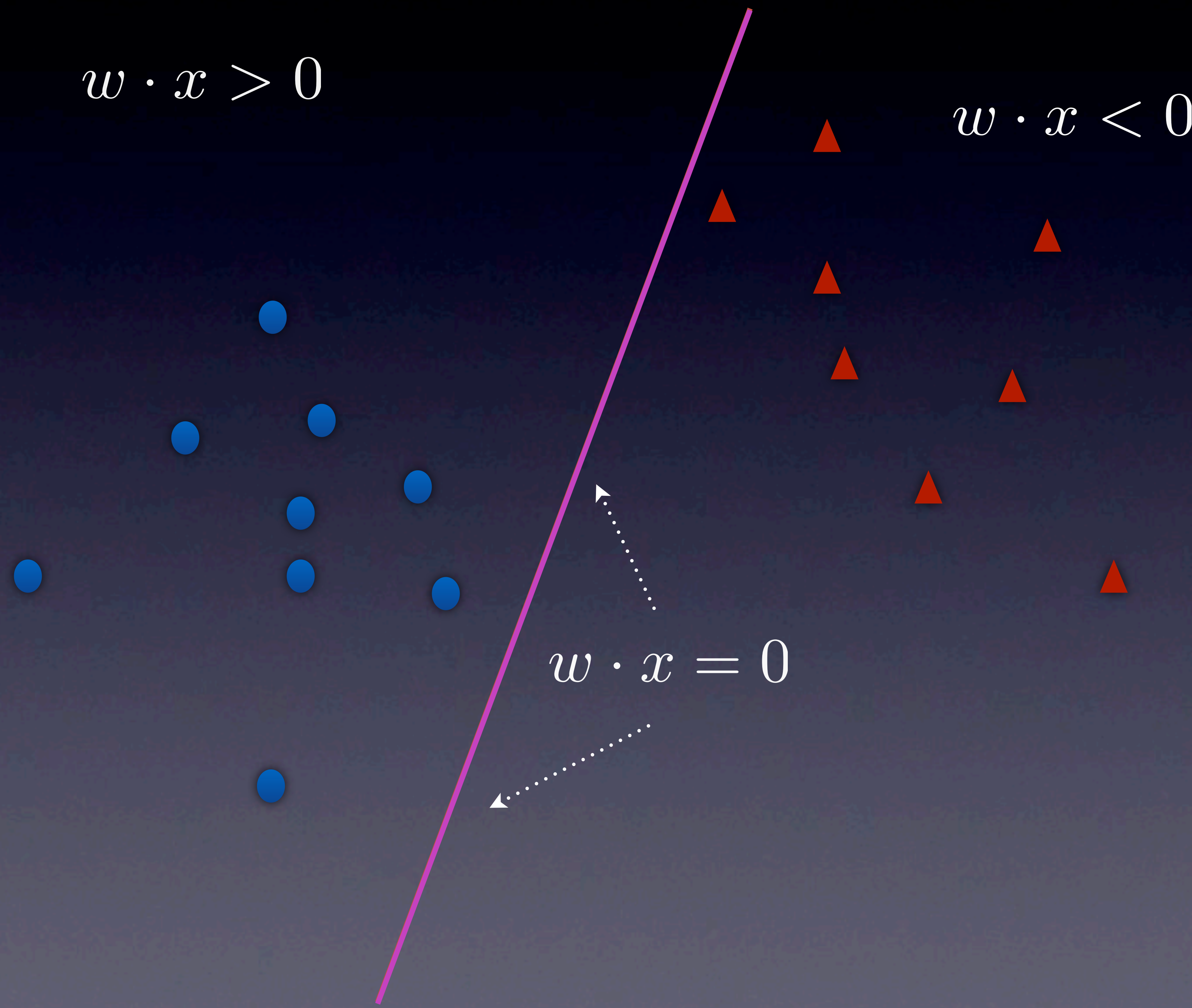


How do we find a good solution?

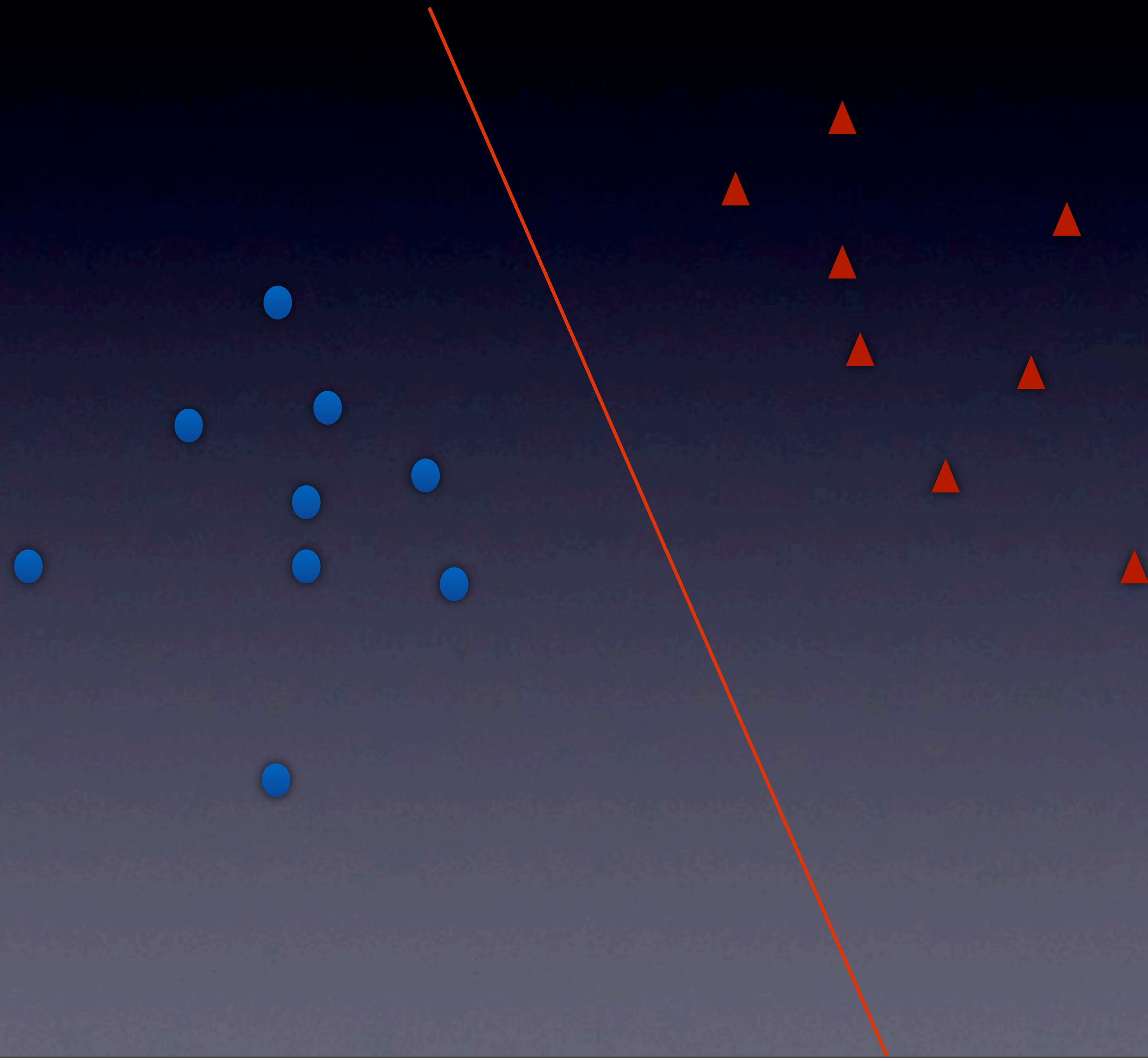
$$x = (x^1, x^2)$$

2D Classification Problem

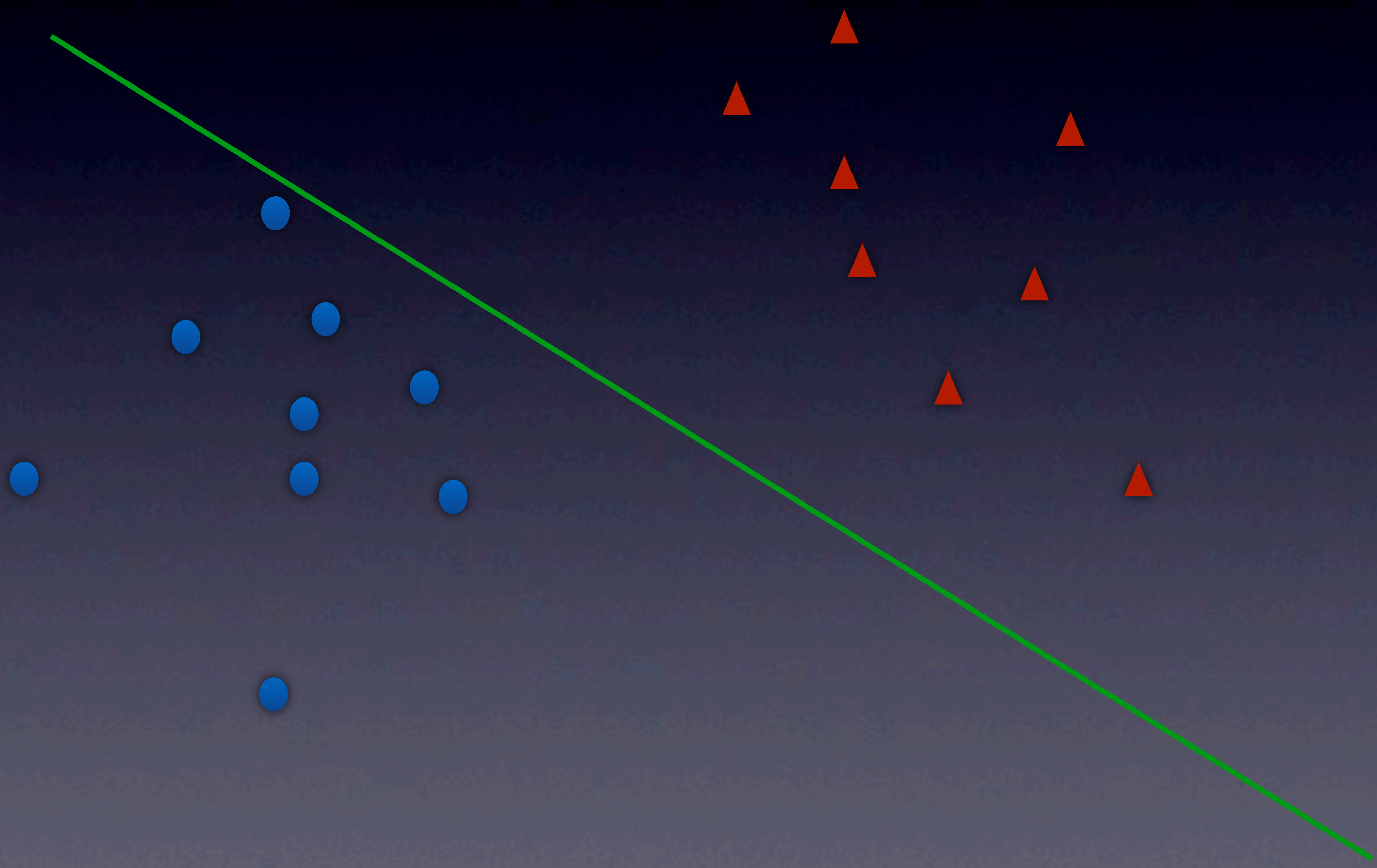
How do we find a good solution?



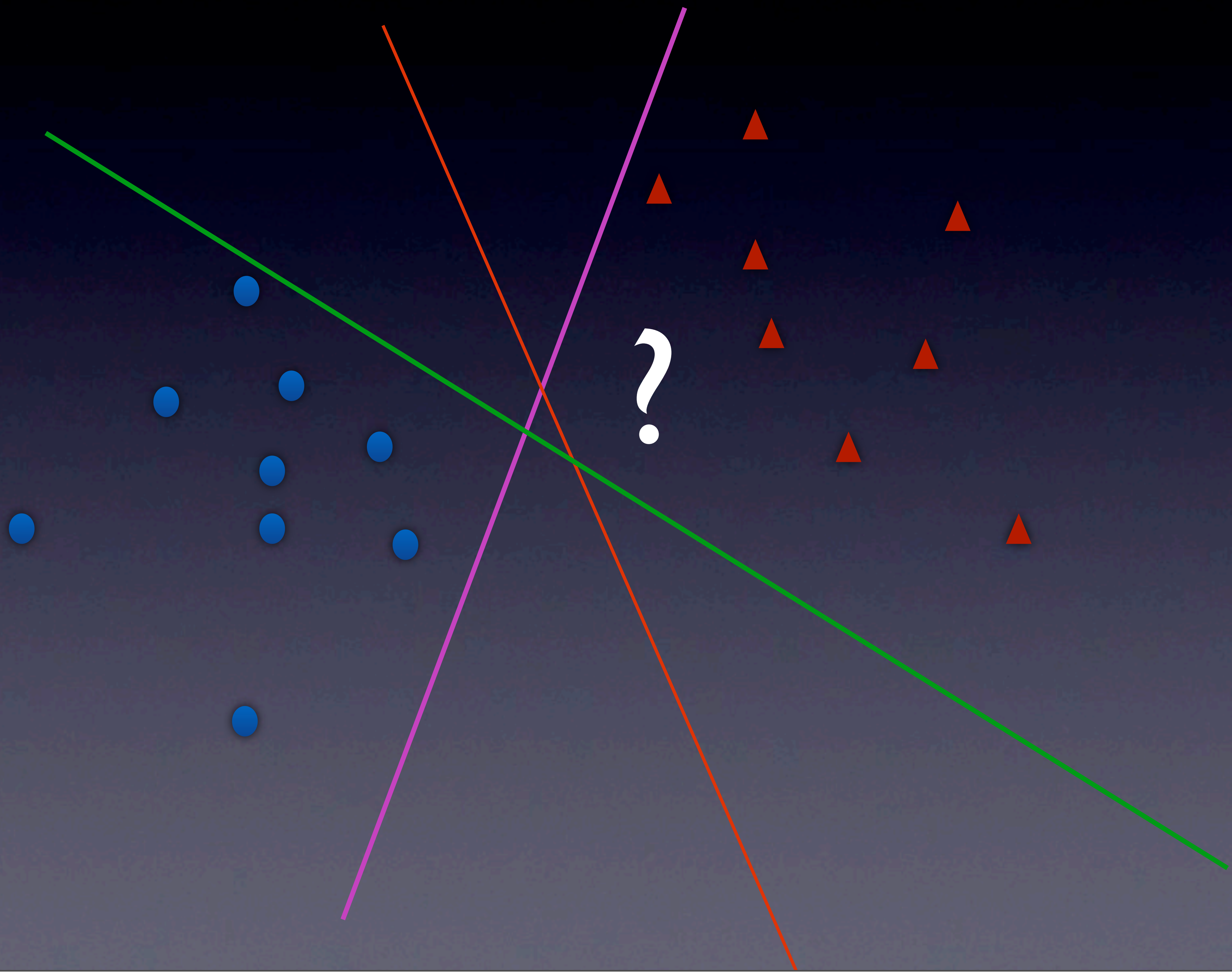
How do we find a good solution?



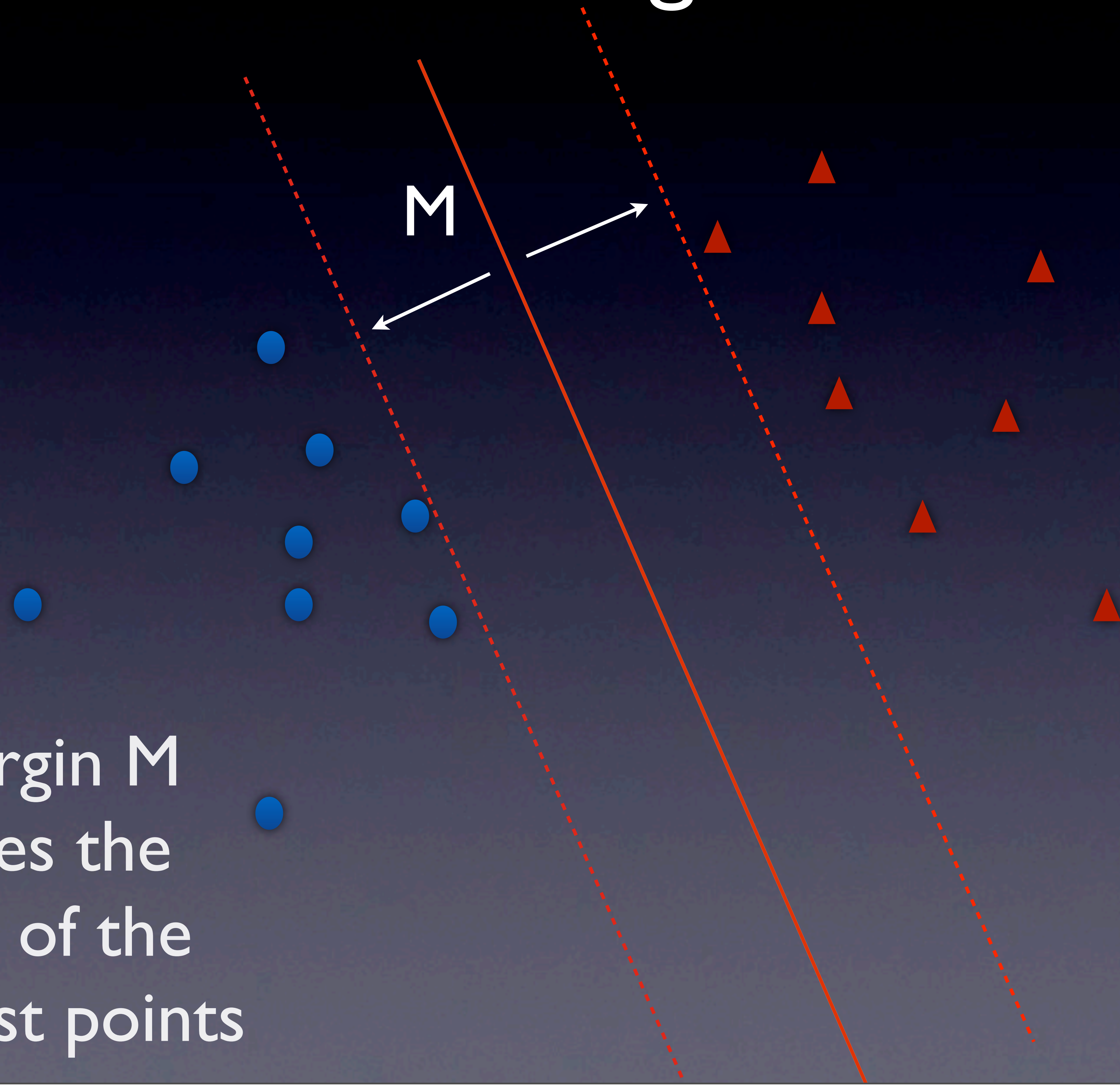
How do we find a good solution?



How do we find a good solution?



How do we find a good solution?



The margin M measures the distance of the two closest points

Maximum Margin Hyperplane

....with little effort ... one can show that

maximizing the margin M is equivalent to:

maximizing

$$\frac{1}{||w||}$$

SVM

Linear and Separable SVM

$$\min_{w \in \mathcal{R}^n} \quad ||w||^2$$
$$\text{subject to :} \quad y_i(w \cdot x) \geq 1 \quad i = 1, \ldots, \ell$$

Typically an off-set term is added to the solution

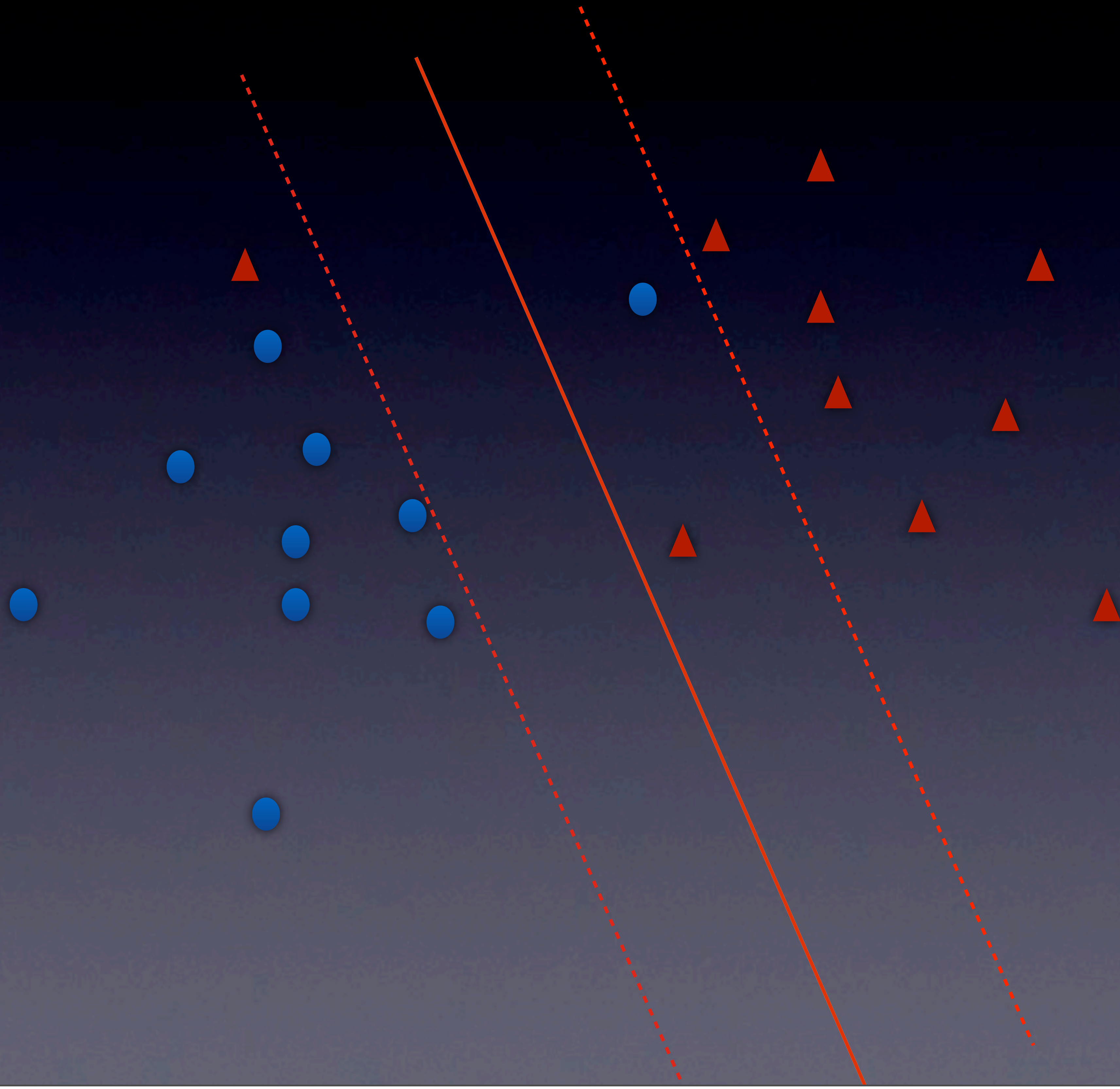
$$f(x) = \text{sign } (w \cdot x + b).$$

A more general Algorithm

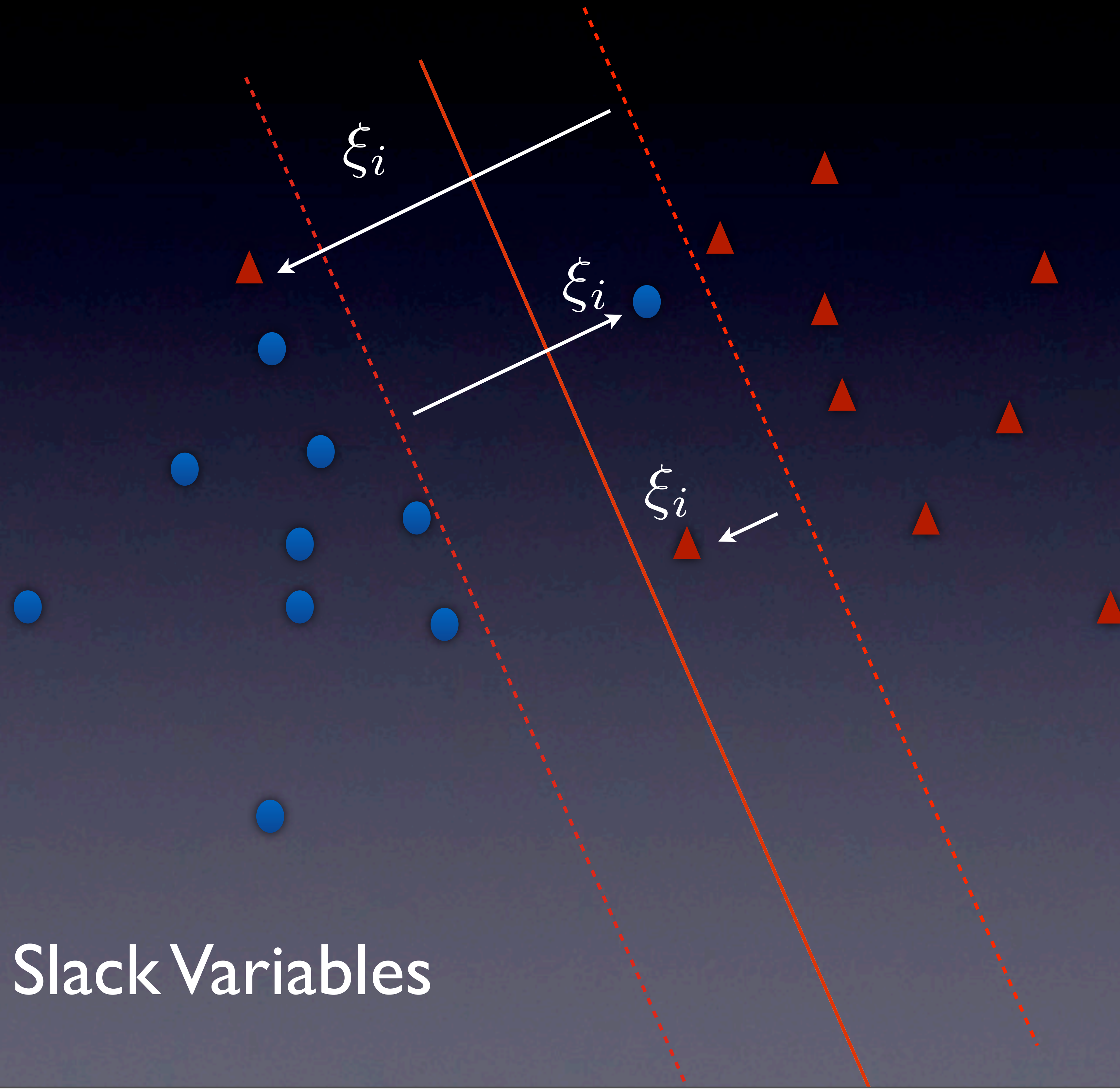
There are two things we would like to improve:

- Allow for errors
- Non Linear Models

Measuring errors



Measuring errors (cont)



ξ_i Slack Variables

Linear SVM

$$
\begin{aligned}
\min_{w \in \mathcal{R}^n, \xi \in \mathcal{R}^n, b \in \mathcal{R}} \quad & C \sum_{i=1}^{\ell} \xi_i + \frac{1}{2} ||w||^2 \\
\text{subject to :} \quad & y_i(w \cdot x + b) \geq 1 - \xi_i \quad i = 1, \ldots, \ell \\
& \xi_i \geq 0 \quad i = 1, \ldots, \ell
\end{aligned}
$$

Optimization

How do we solve this minimization problem?

(...and why do we call it SVM anyway?)

Some facts

- Representer Theorem
- Dual Formulation
- Box Constraints and Support Vectors

Representer Theorem

The solution to the minimization problem can be written as

$$w \cdot x = \sum_{i=1}^{\ell} c_i (x \cdot x_i)$$

Dual Problem

The coefficients can be found solving:

$$\max_{\alpha \in \mathcal{R}^\ell} \quad \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\alpha^T Q \alpha$$

$$\text{subject to :} \qquad \sum_{i=1}^{\ell} y_i \alpha_i = 0$$

$$0 \leq \alpha_i \leq C \qquad i = 1, \ldots, \ell$$

Here $Q = y_i y_j (x_i \cdot x_j)$

$$\alpha_i = c_i / y_i$$

Optimality conditions

with little effort ... one can show that

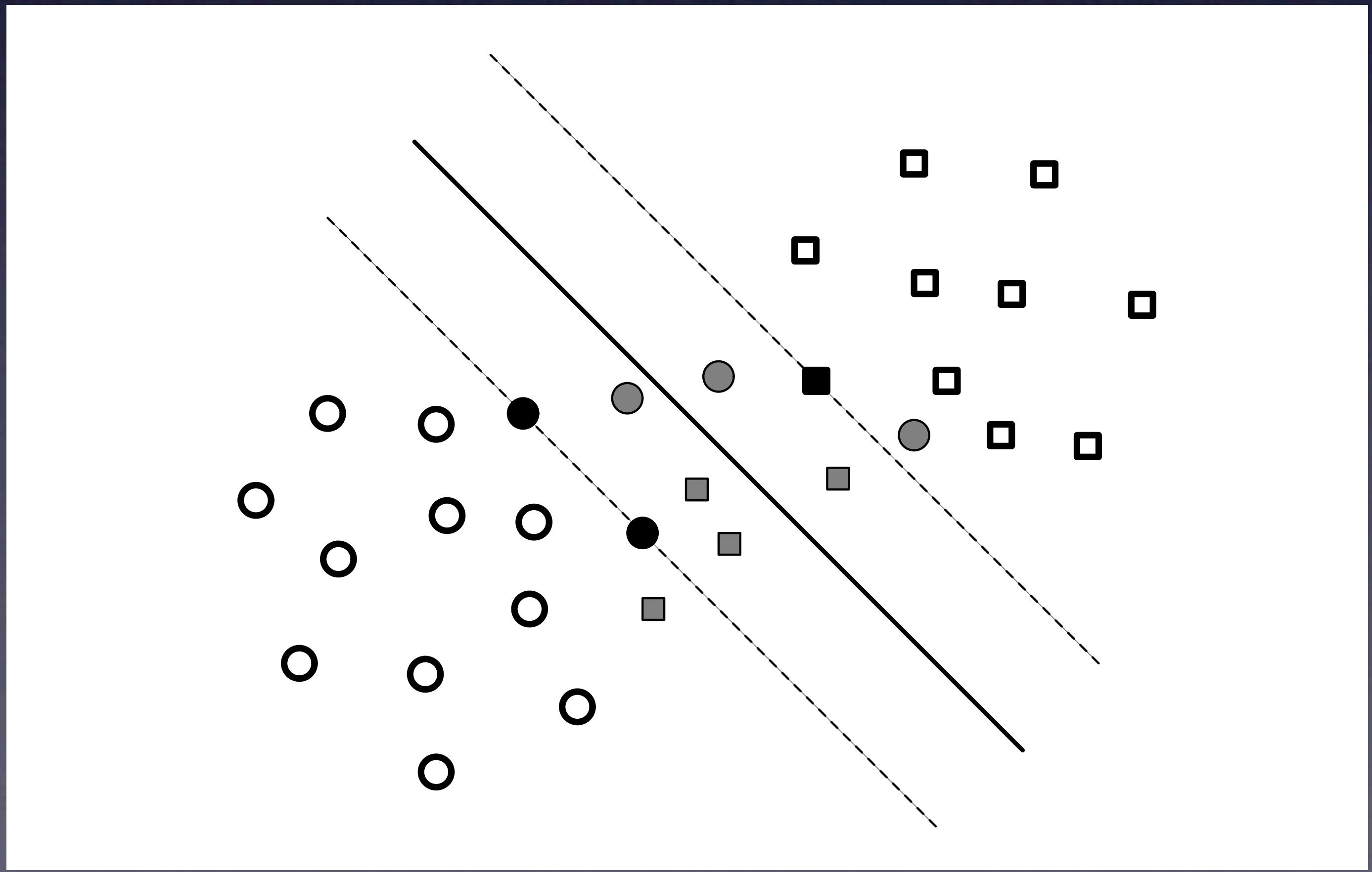
If $\boxed{\alpha_i > 0}$ then $\boxed{y_i f(x_i) \leq 1}$

The solution is *sparse*: some training points do not contribute to the solution.

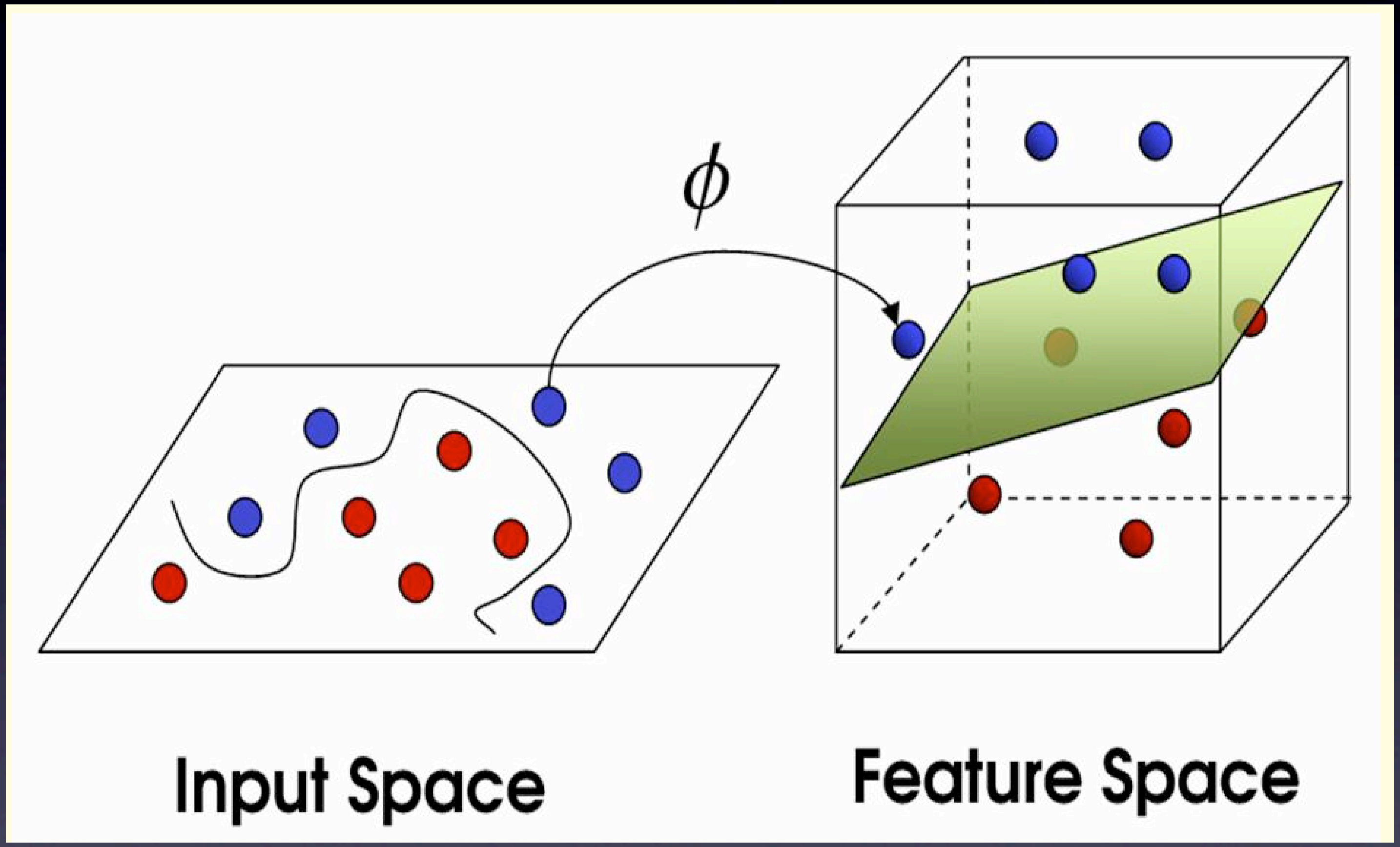
Sparse Solution

Note that:

The solution depends only on the training set points. (no dependence on the number of features!)



Feature Map



$$f(x) = w \cdot \Phi(x)$$

A Key Observation

The solution depends only on $Q = y_i y_j (x_i \cdot x_j)$

$$\max_{\alpha \in \mathcal{R}^\ell} \quad \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\alpha^T Q \alpha$$

$$\text{subject to :} \quad \sum_{i=1}^{\ell} y_i \alpha_i = 0$$

$$0 \leq \alpha_i \leq C \qquad i = 1, \ldots, \ell$$

Idea: use $Q = y_i y_j (\Phi(x_i) \cdot \Phi(x_j))$

Kernels and Feature Maps

The crucial quantity is the inner product

$$K(x,t) = \Phi(x) \cdot \Phi(t)$$

called *Kernel.*

A function is called Kernel if it is:

- **symmetric**
- **positive definite**

Examples of Kernels

- **Linear kernel**

$$K(x, x') = x \cdot x'$$

- **Gaussian kernel**

$$K(x, x') = e^{-\frac{\|x - x'\|^2}{\sigma^2}}, \qquad \sigma > 0$$

- **Polynomial kernel**

$$K(x, x') = (x \cdot x' + 1)^d, \qquad d \in \mathbb{N}$$

For specific applications, designing an effective kernel is a challenging problem.

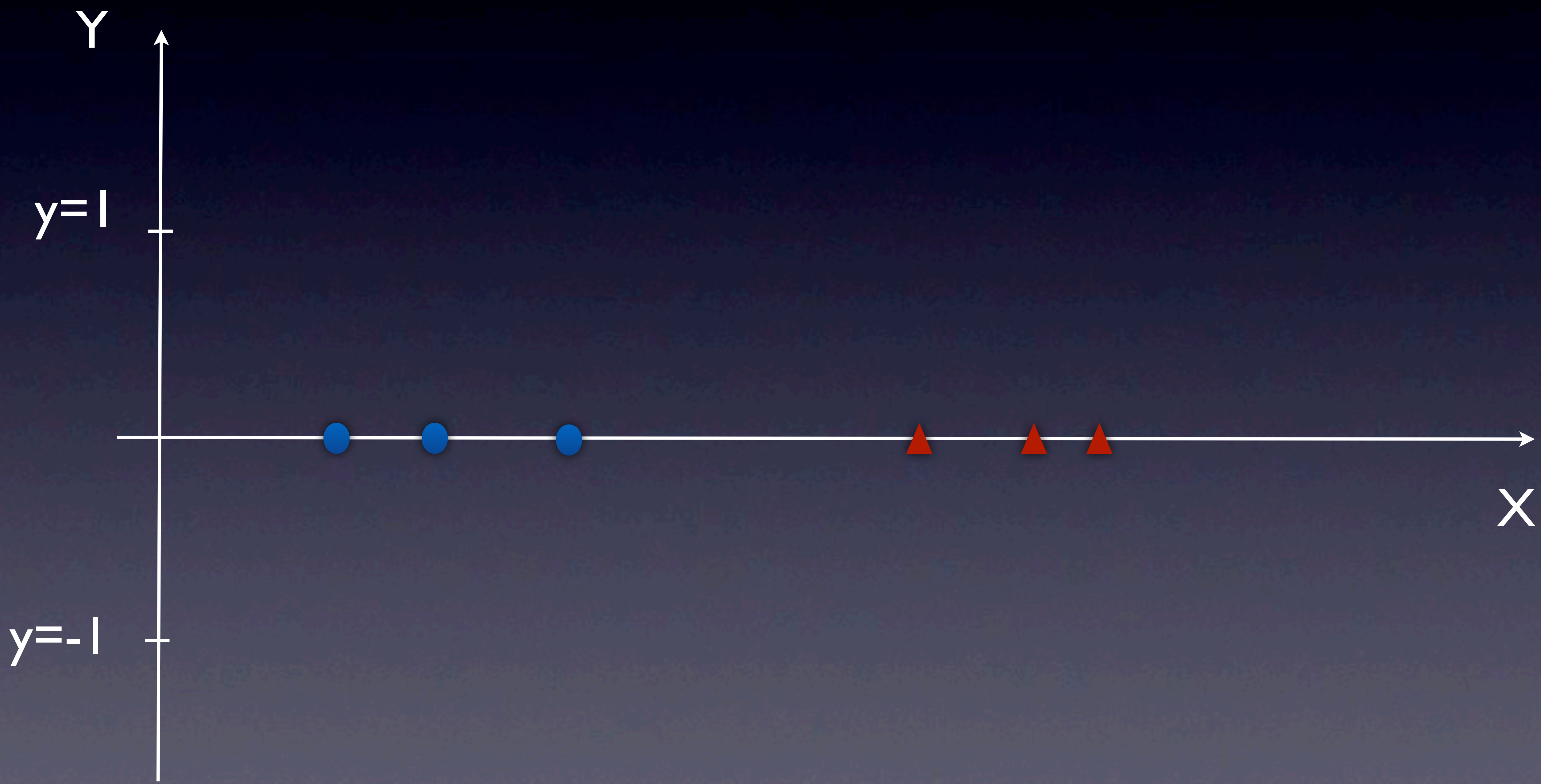
Non Linear SVM

Summing up:

- Define Feature Map either explicitly or via a kernel
- Find linear solution in the Feature space
- Use same solver as in the linear case
- Representer theorem now gives:

$$w \cdot \Phi(x) = \sum_{i=1}^{\ell} c_i (\Phi(x) \cdot \Phi(x_i)) = \sum_{i=1}^{\ell} c_i K(x, x_i)$$

Example in 1D



Software

- SVM Light: `http://svmlight.joachims.org`
- SVM Torch: `http://www.torch.ch`
- libSVM: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Model Selection

- We have to fix the Regularization parameter C
- We have to choose the kernel (and its parameter)

Using default values is usually a BAD BAD idea

Regularization Parameter

$$\min_{w \in \mathcal{R}^n, \xi \in \mathcal{R}^n, b \in \mathcal{R}} \quad C \sum_{i=1}^{\ell} \xi_i + \frac{1}{2} ||w||^2$$

- Large C: we try to minimize errors ignoring the complexity of the solution
- Small C we ignore the errors to obtain a simple solution

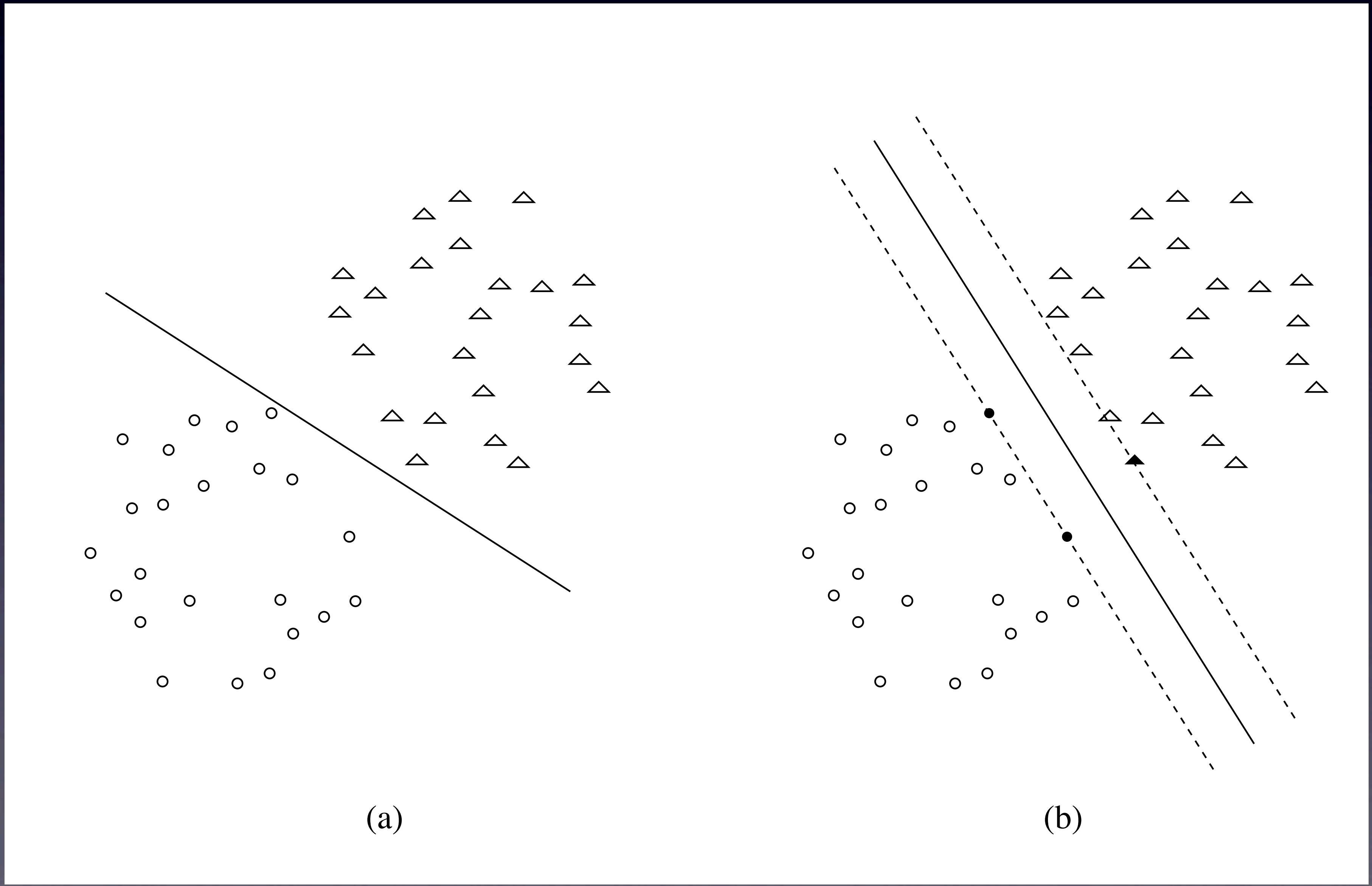
Which Kernel?

- For very high dimensional data linear kernel is often the default choice
  - allows computational speed up
  - less prone to overfitting
- Gaussian Kernel with proper tuning is another common choice

Whenever possible use prior knowledge to build problem specific features or

2D demo

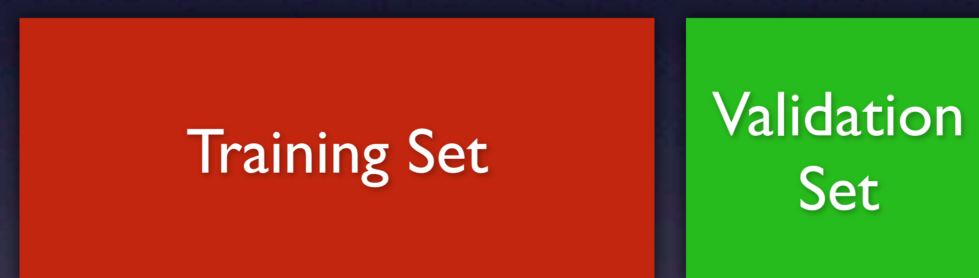
demo



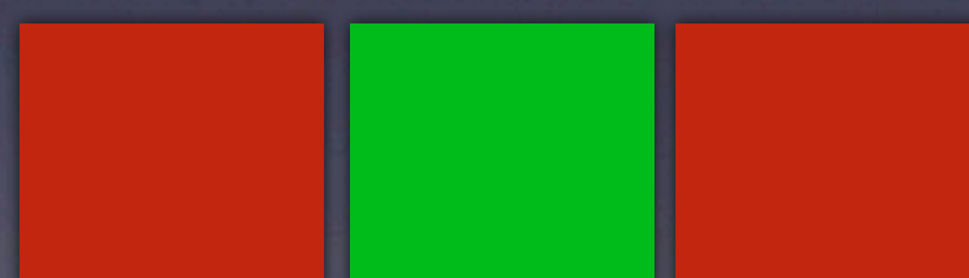
Practical Rules

We can choose C (and the kernel parameter) via cross validation

- Holdout set



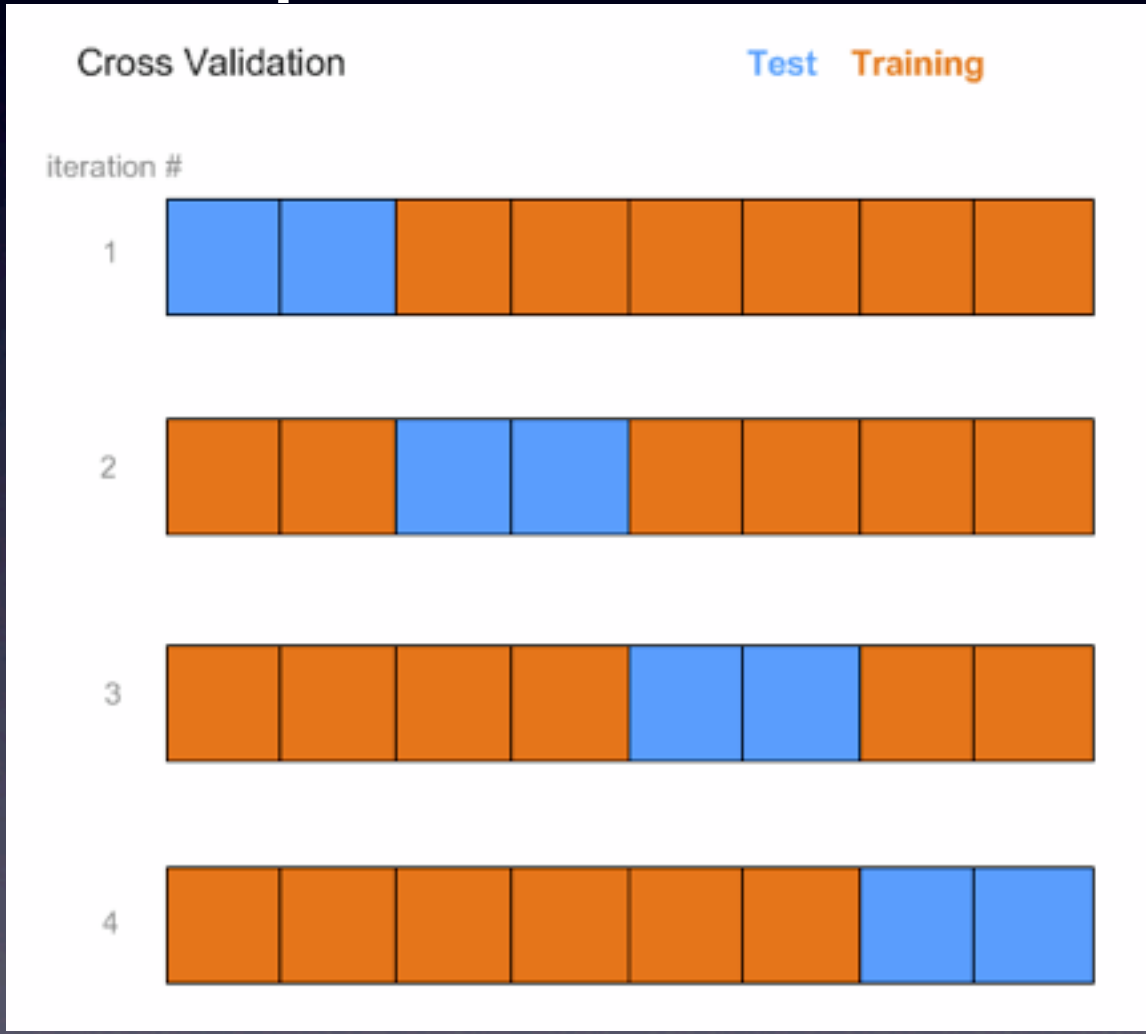
- K-fold cross validation



- K=# of examples is called Leave One Out

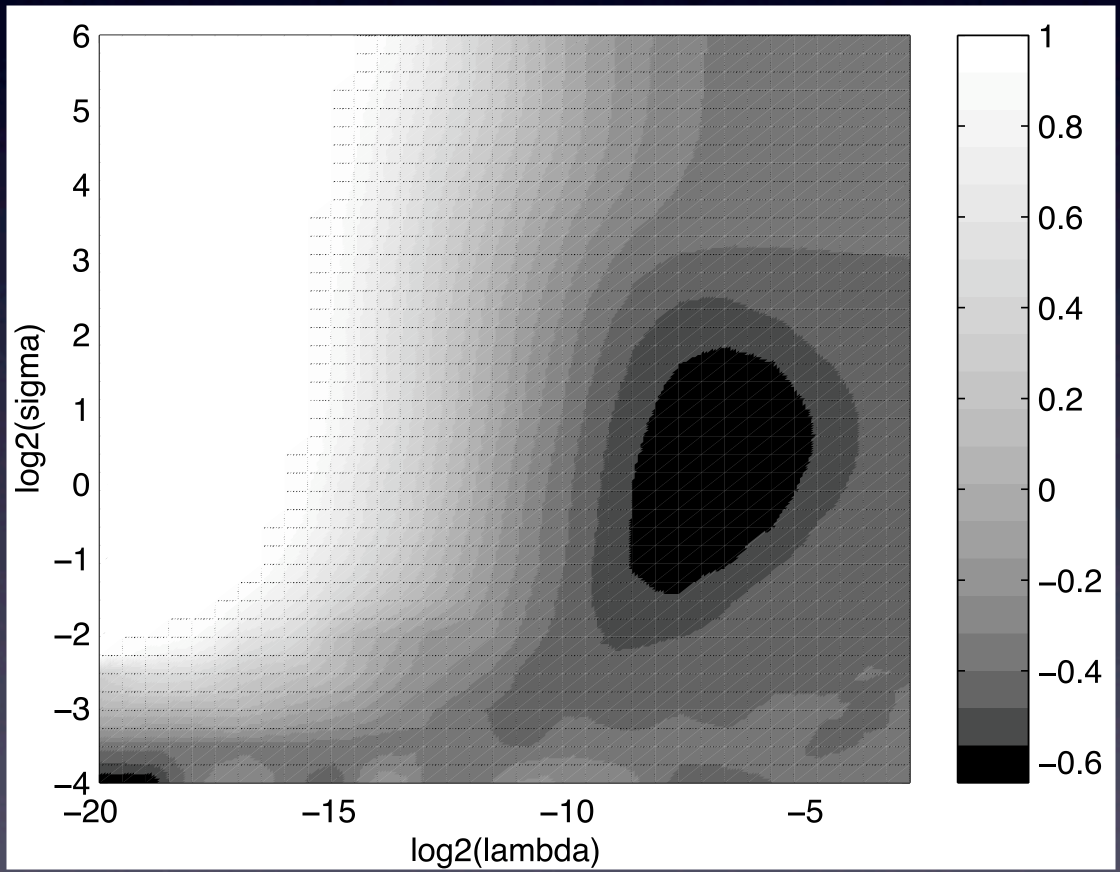
K-Fold CV

We have to compute several solutions...



A Rule of Thumb

This is how the CV error typically looks like



Fix a reasonable kernel, then fine tune C

Which values do we start from?

- For the Gaussian kernel, pick sigma of the order of the average distance...

$$k(X_i, X_j) = \exp\left(-\frac{||X_i - X_j||^2}{\sigma^2}\right)$$

- Take min (and max) C as the value for which the training set error does not increase (decrease) anymore.

Computational Considerations

- the training time depends on the parameters: the more we fit, the slower the algorithm.
- typically the computational burden is in the selection of the regularization parameter (*solvers for regularization path*).

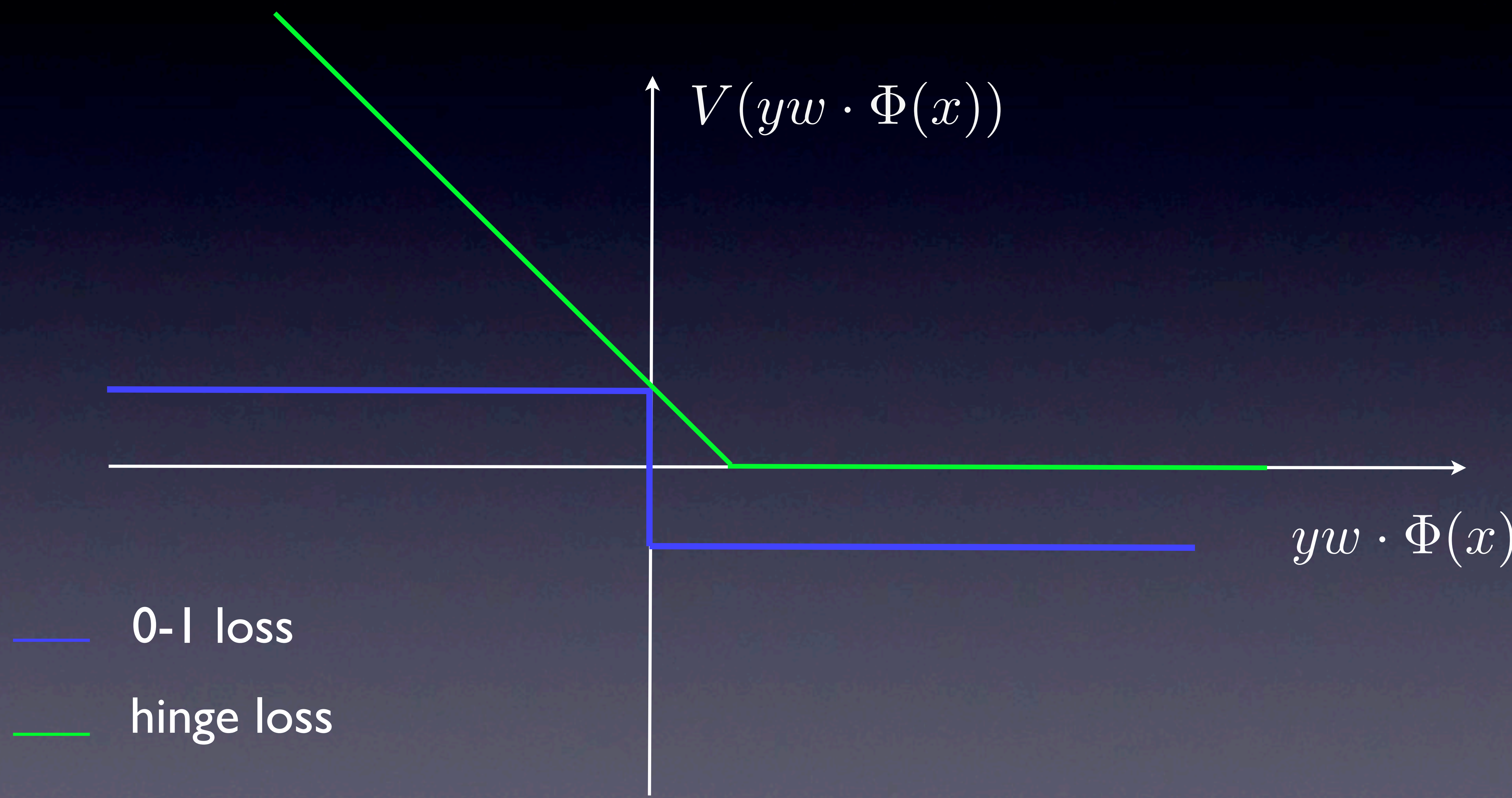
Regularization Networks

SVM are an example of a family of algorithms of the form:

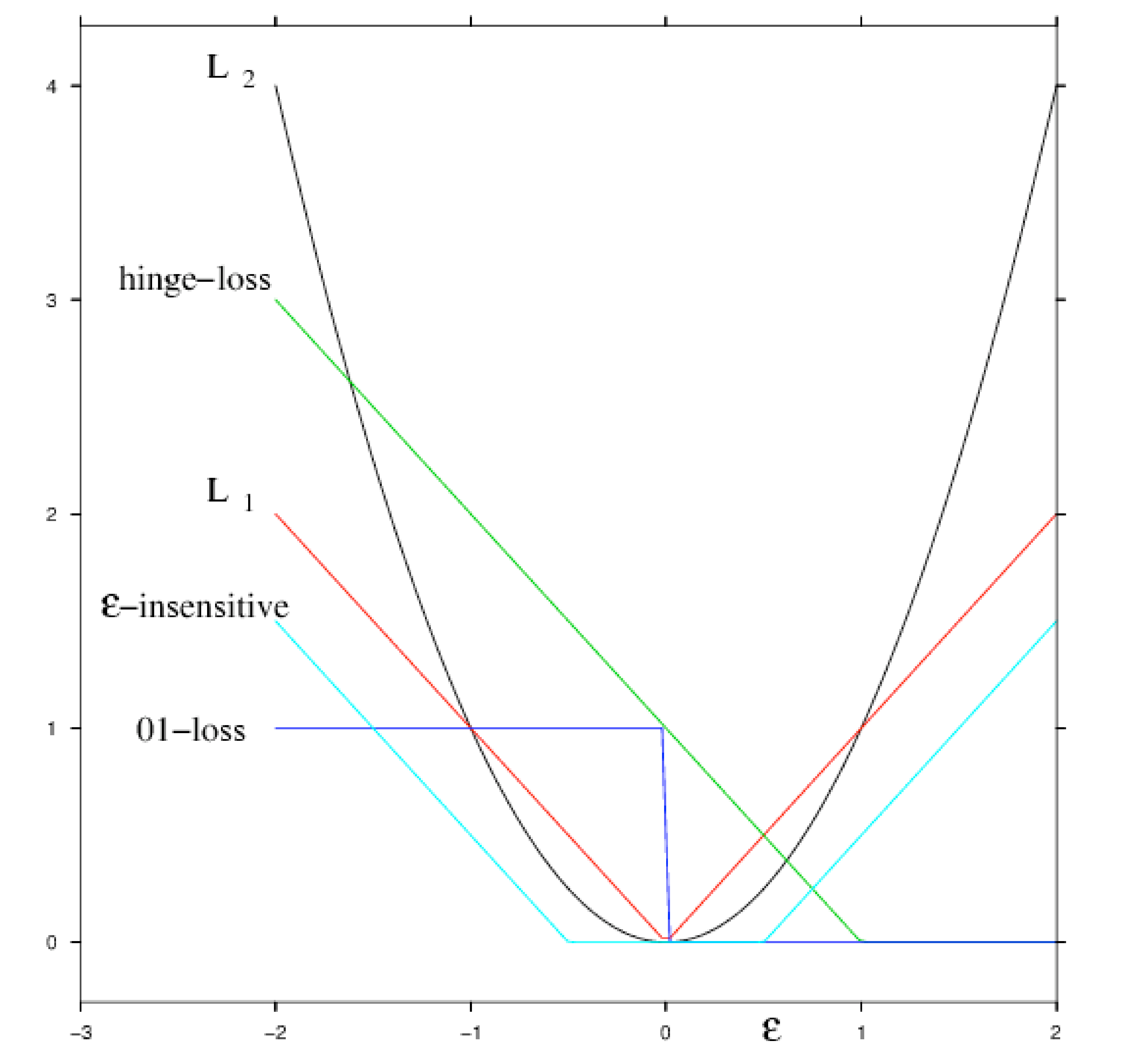
$$C\sum_{i=1}^{\ell} V(y_i, w\cdot\Phi(x_i)) + \|w\|^2$$

V is called loss function

Hinge Loss



Loss functions



Representer Theorem

For a LARGE class of loss functions:

$$w \cdot \Phi(x) = \sum_{i=1}^{n} \alpha_i (\Phi(x) \cdot \Phi(x_i)) = \sum_{i=1}^{n} \alpha_i K(x, x_i)$$

The way we compute the coefficients depends on the considered loss function.

Regularized LS

The simplest, yet powerful, algorithm is probably RLS

Square loss $V(y, w \cdot \Phi(x)) = (y - w \cdot \Phi(x))^2$

Algorithm $(Q + \frac{1}{C} I)\alpha = y, \quad Q_{i,j} = K(x_i, x_j)$

Leave one out can be computed at the price of one (!!!) solution

Summary

- Separable, Linear SVM
- Non Separable, Linear SVM
- Non Separable, Non Linear SVM
- How to use SVM