Support Vector Machines

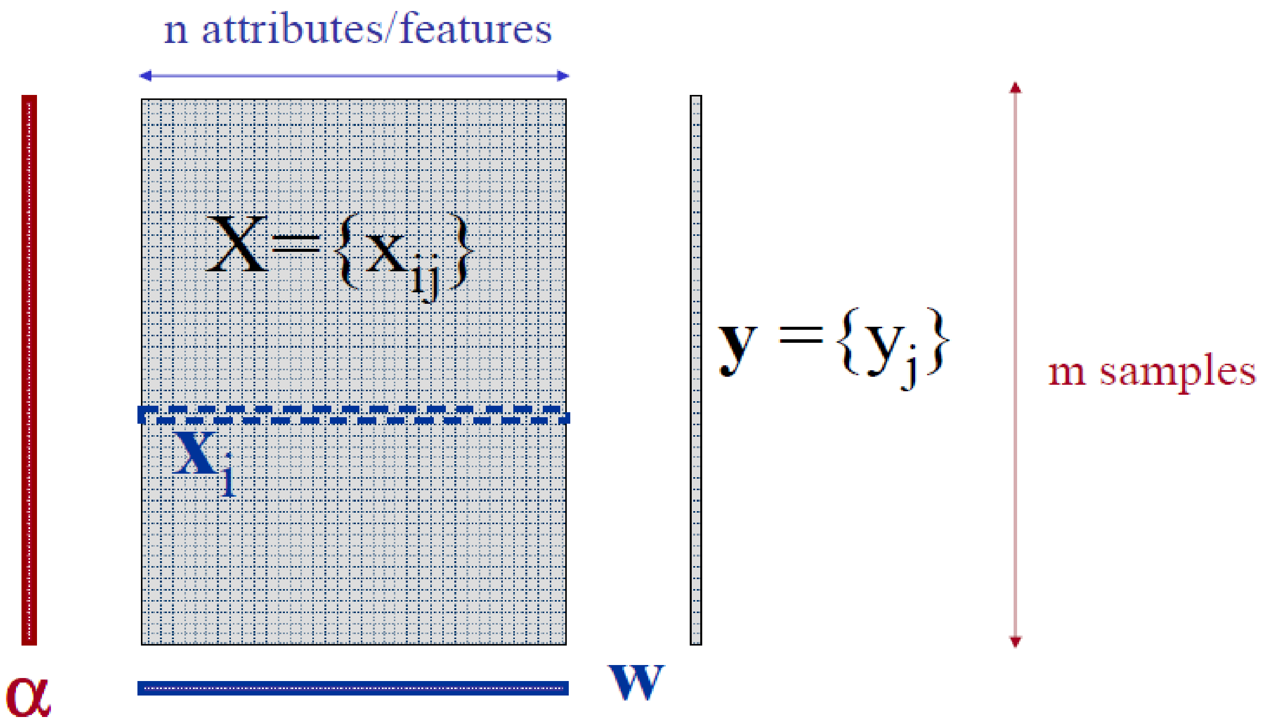
Dr. Saed Sayad
University of Toronto
2010
saed.sayad@utoronto.ca

SVM
A New Generation of Learning Algorithms

- Pre 1980
  - Almost all learning methods learned linear decision surfaces.
  - Linear learning methods have nice theoretical properties.
- 1980’s
  - Decision trees and Neural Networks allowed efficient learning of non-linear decision surfaces.
  - Little theoretical basis and all suffer from local minima.
- 1990’s
  - Developing efficient learning algorithms for non-linear functions based on computational learning theory.
  - Robust theoretical properties.

SVM and Linear Regression

$$y = \sum_{i=1}^{m} \alpha_i k(x_i x) + b$$

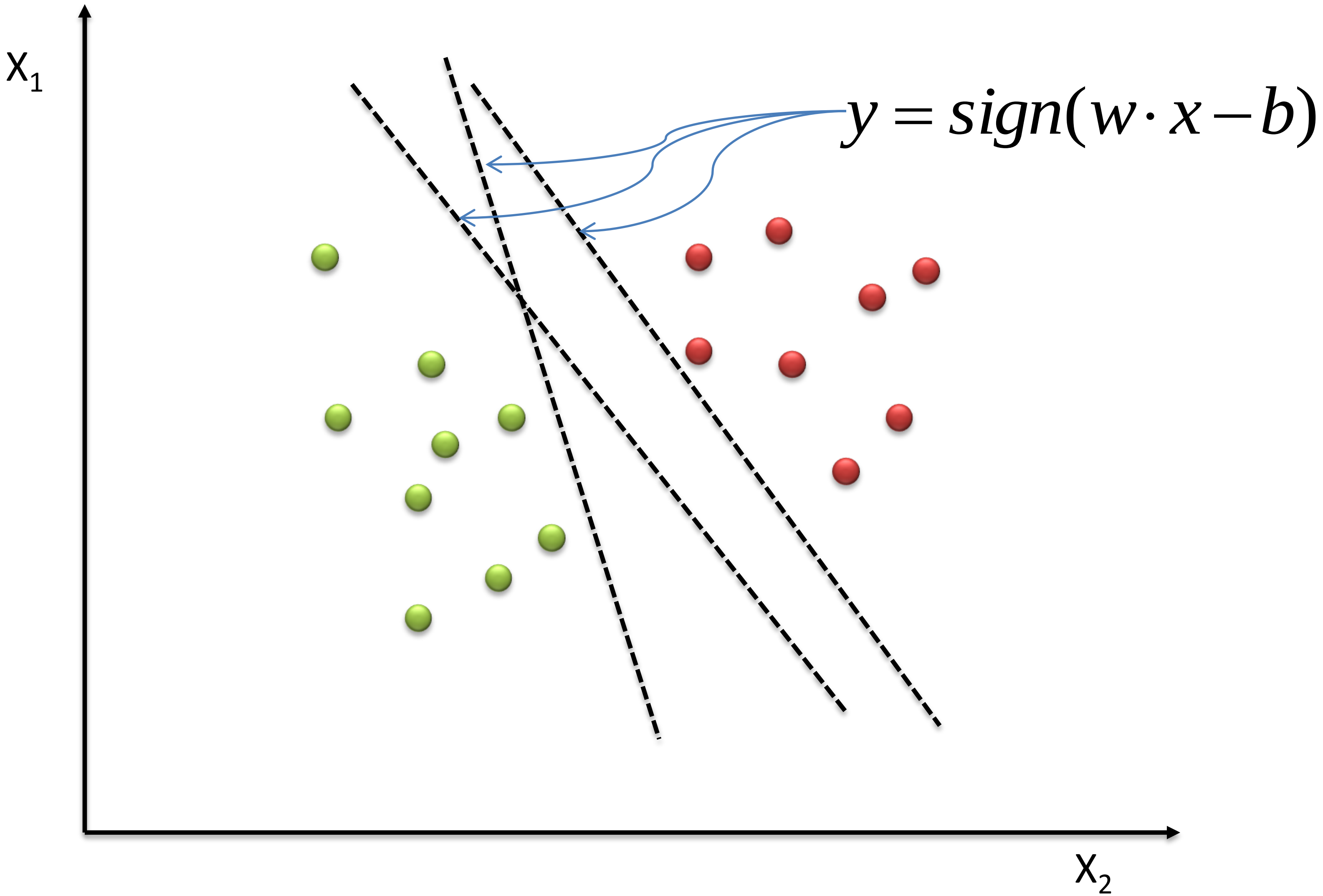


$$y = \sum_{j=1}^{n} w_j x_j + b$$

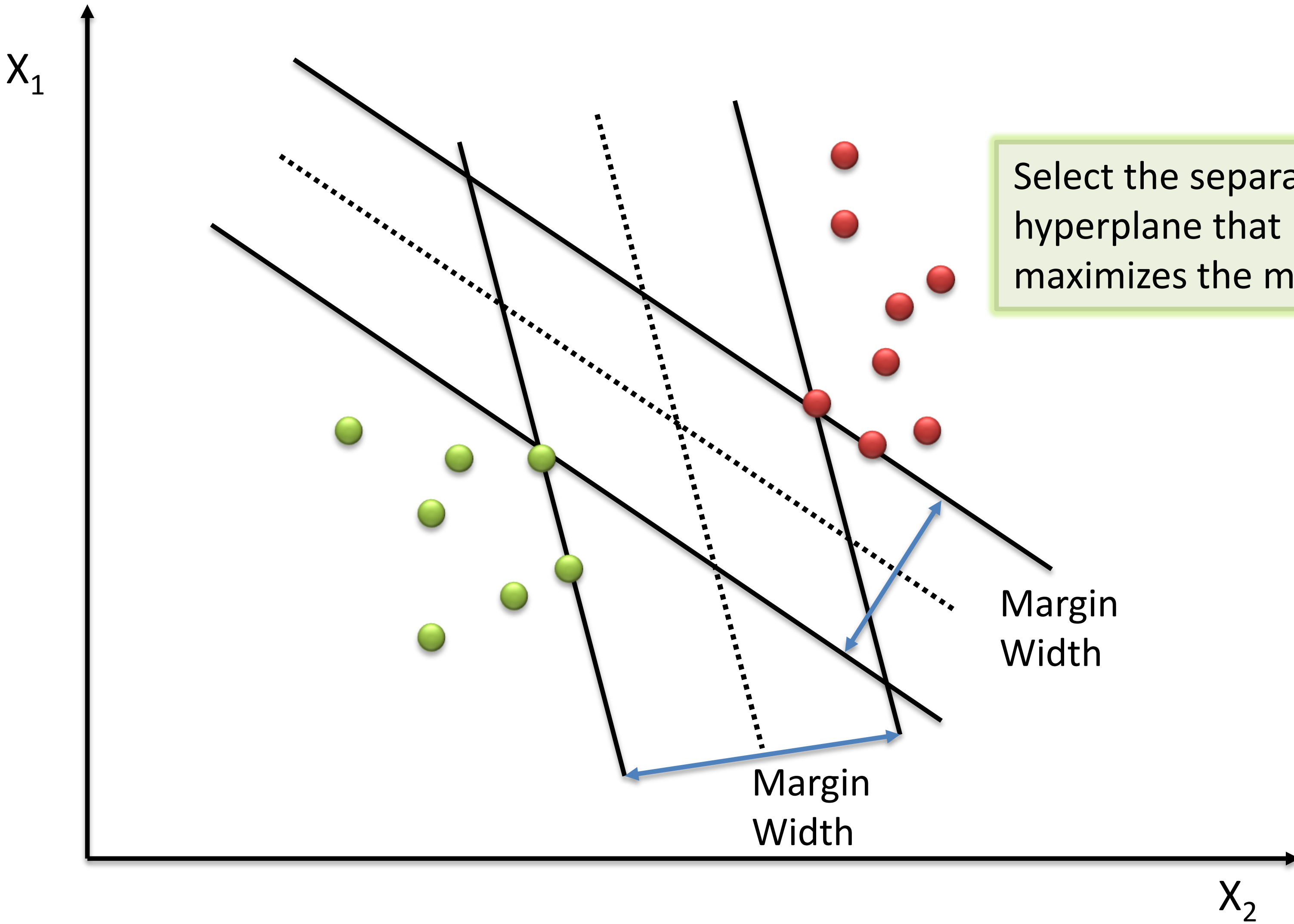
Support Vector Machines - Ideas

- Three main ideas:
  1. Define an optimal hyperplane: **maximize margin**
  2. Extend the above definition for non-linearly separable problems: **have a penalty term for misclassifications.**
  3. Map data to high dimensional space where it is easier to classify with linear decision surfaces: **reformulate problem so that data is mapped implicitly to this space**.

How would you classify this data?

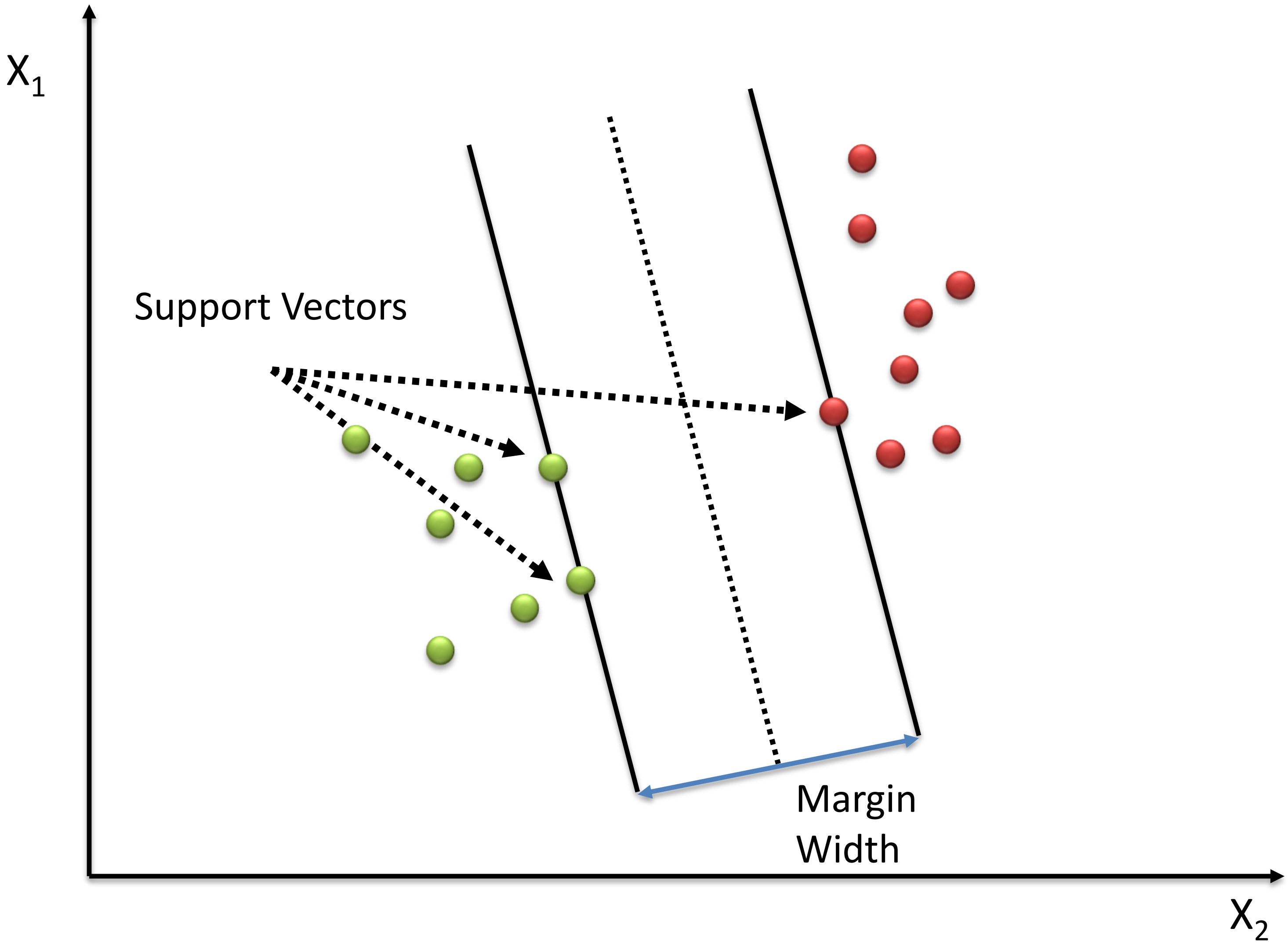


Maximizing the Margin

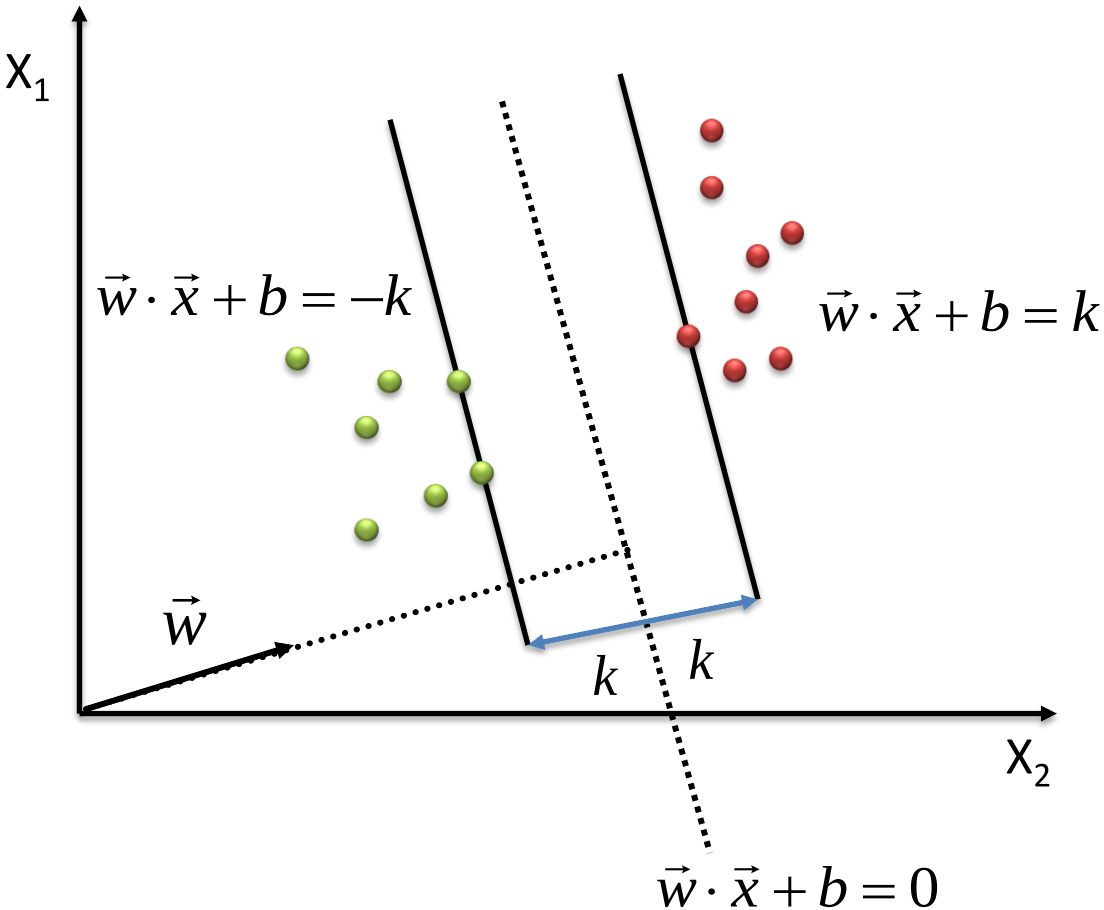


Select the separating hyperplane that maximizes the margin.

Support Vectors



Optimization Problem



Optimization problem is maximizing the width of the margin

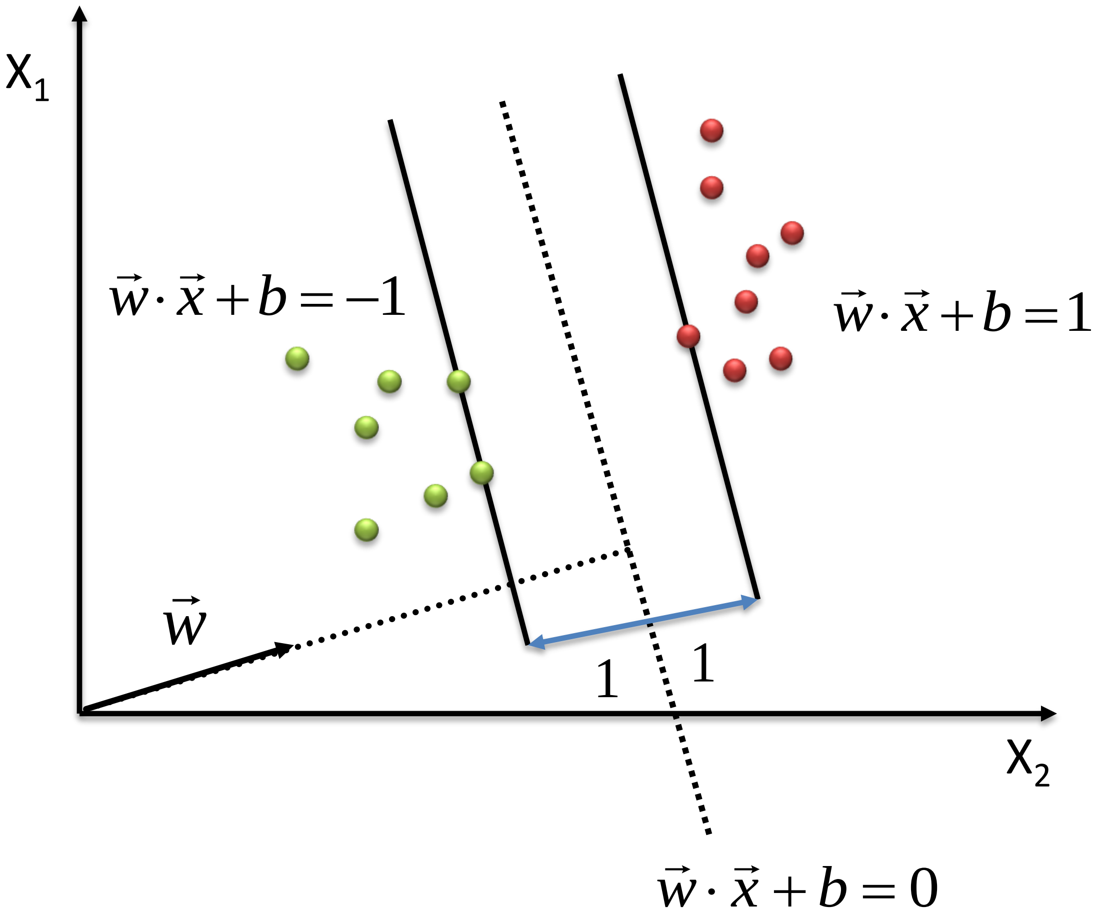
$$\max \frac{k}{\|w\|}$$

$s.t.$

$$(w\cdot x+b)\geq k, \forall x \text{ of class } 1$$

$$(w\cdot x+b)\leq -k, \forall x \text{ of class } 2$$

Optimization Problem



There is a scale and unit for data so that *k=1*. Then problem becomes:

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

$$s.t.$$

$$(w\cdot x+b)\geq 1, \forall x \text{ of class } 1$$

$$(w\cdot x+b)\leq -1, \forall x \text{ of class } 2$$

Setting Up the Optimization Problem

- If class 1 corresponds to 1 and class 2 corresponds to -1, we can rewrite

$$(w \cdot x_i + b) \geq 1, \ \forall x_i \text{ with } y_i = 1$$
$$(w \cdot x_i + b) \leq -1, \ \forall x_i \text{ with } y_i = -1$$

- as

$$y_i (w \cdot x_i + b) \geq 1, \ \forall x_i$$

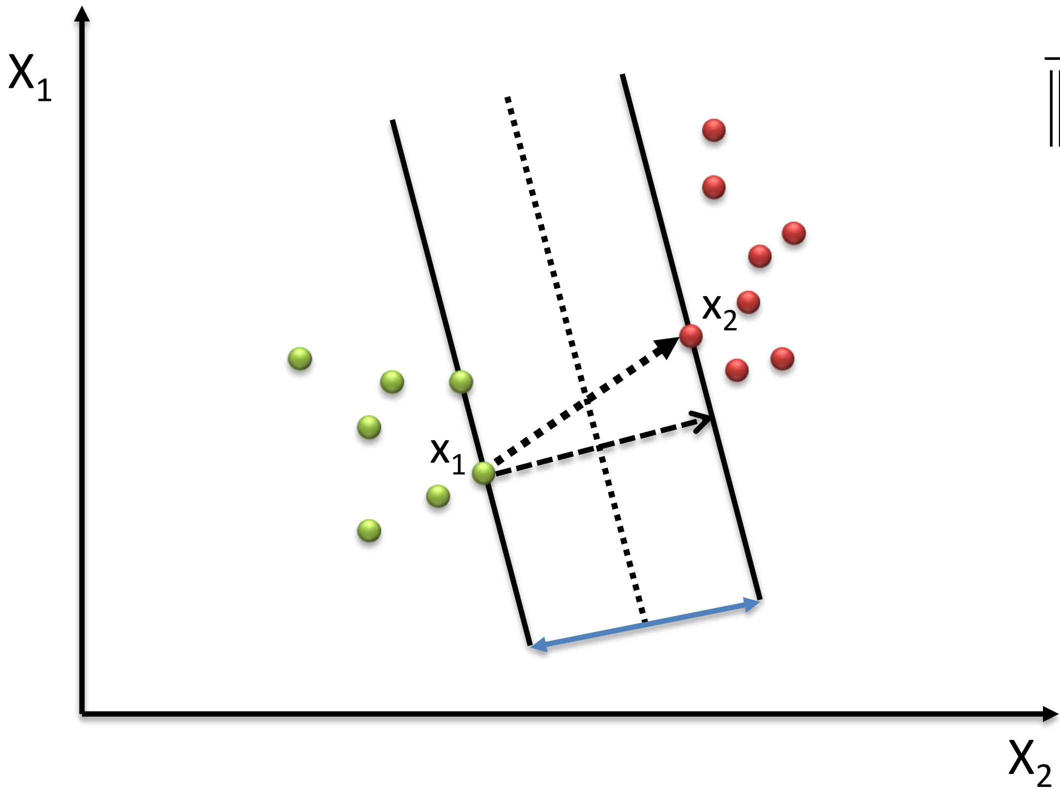
- So the problem becomes:

$$\max \frac{2}{\|w\|} \quad s.t.\ y_i (w \cdot x_i + b) \geq 1, \ \forall x_i$$

or

$$\min \frac{1}{2} \|w\|^2 \quad s.t.\ y_i (w \cdot x_i + b) \geq 1, \ \forall x_i$$

Margin Width



$$\frac{w}{\|w\|}\cdot(x_2 - x_1) = \text{width} = \frac{2}{\|w\|}$$

$$w \cdot x_2 + b = 1$$

$$w \cdot x_1 + b = -1$$

$$w \cdot x_2 + b - w \cdot x_1 - b = 1 - (-1)$$

$$w \cdot x_2 - w \cdot x_1 = 2$$

$$\frac{w}{\|w\|}(x_2 - x_1) = \frac{2}{\|w\|}$$

Linear, Hard-Margin SVM Formulation

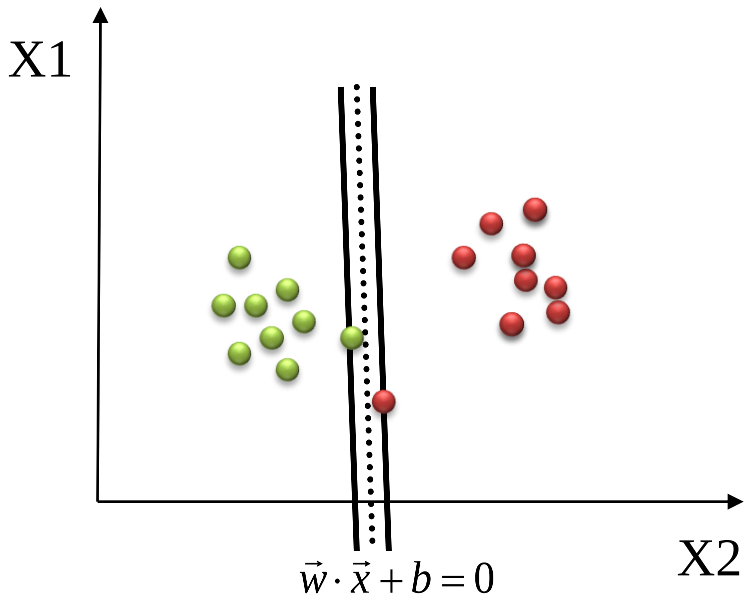
- Find *w, b* that solves

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

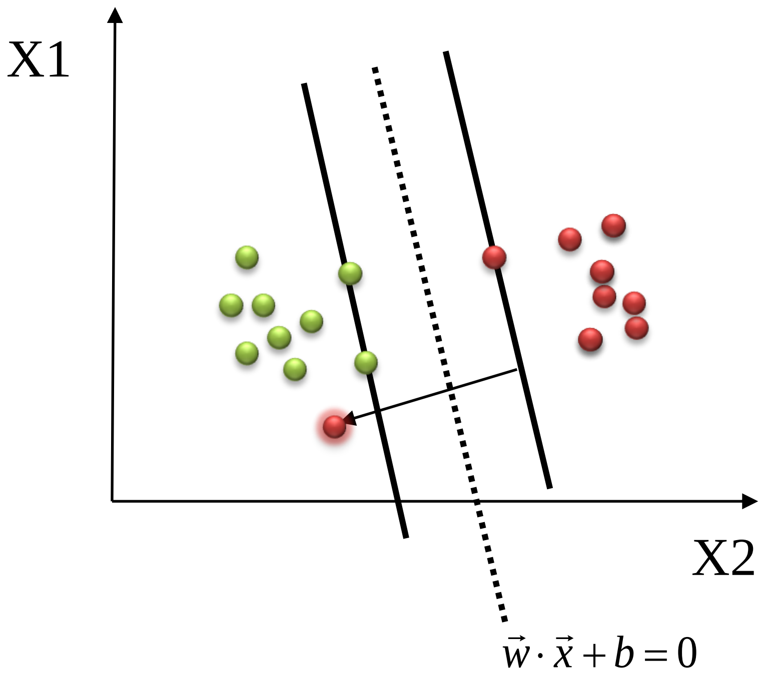
$$s.t.\ y_i(w \cdot x_i + b) \geq 1,\ \forall x_i$$

- Problem is convex so, there is a unique global minimum value (when feasible).
- Non-solvable if the data is not linearly separable
- **Quadratic Programming**

Soft vs Hard Margin SVMs

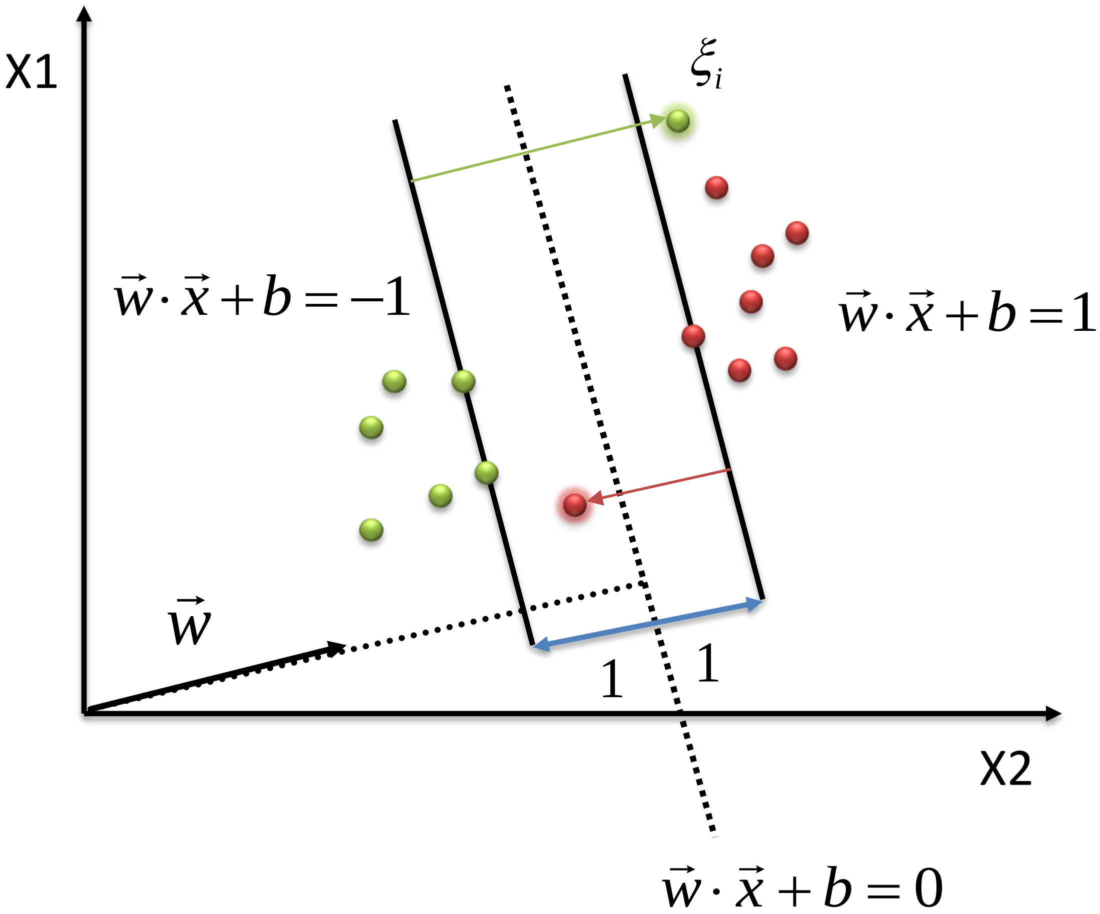


Hard Margin SVM



Soft Margin SVM

Non-Linearly Separable Data

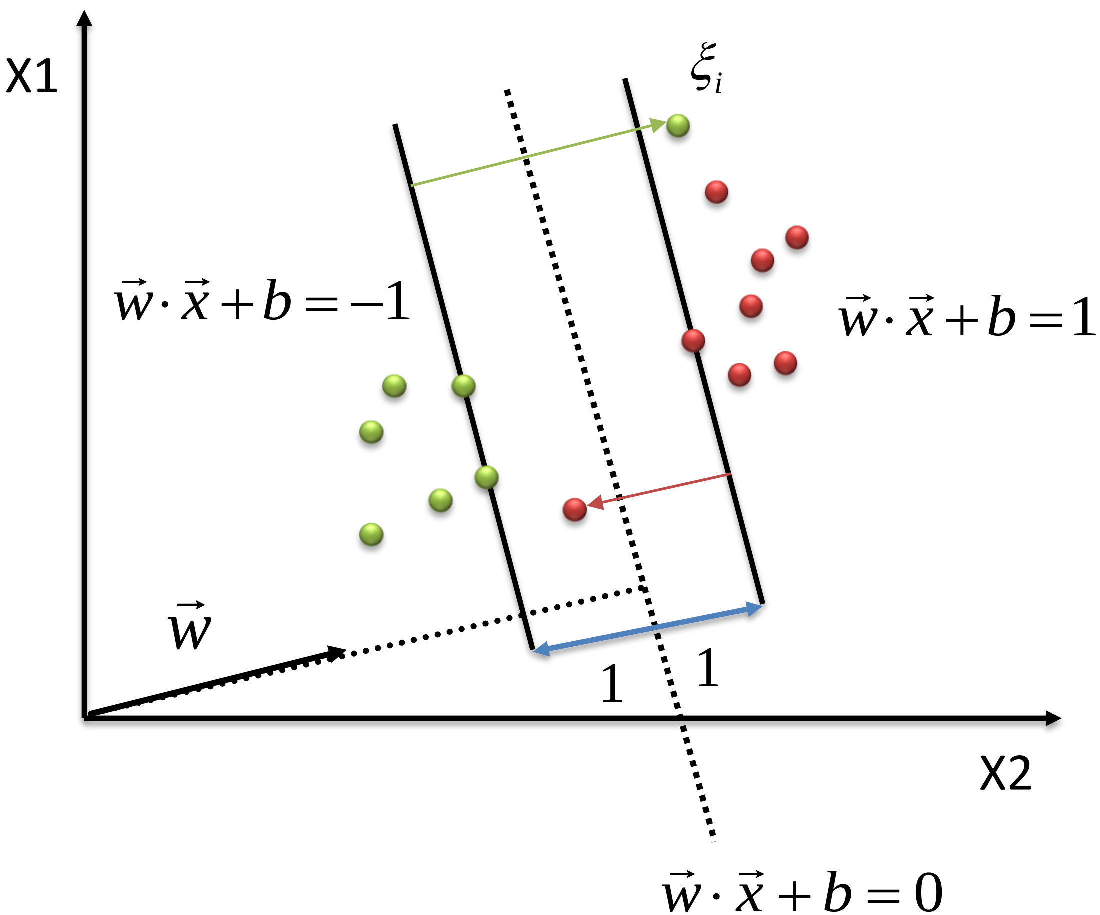


slack variable:

$$\xi_i$$

Allow some instances to fall off the margin, but penalize them

Formulating the Optimization Problem



Constraint becomes :

$$y_i(w\cdot x_i+b)\geq 1-\xi_i,\ \forall x_i$$
$$\xi_i\geq 0$$

Objective function penalizes for misclassified instances and those within the margin

$$\min\frac{1}{2}\|w\|^2+C\sum_i\xi_i$$

C trades-off margin width and misclassifications

Linear, Soft-Margin SVMs

$$\min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \qquad y_i(w \cdot x_i + b) \geq 1 - \xi_i, \ \forall x_i$$
$$\xi_i \geq 0$$

- Algorithm tries to maintain ξ_i to zero while maximizing margin
- Notice: algorithm does not minimize the *number* of misclassifications (NP-complete problem) but the sum of distances from the margin hyperplanes
- Other formulations use ξ_i^2 instead
- As $C \to \infty$, we get closer to the hard-margin solution

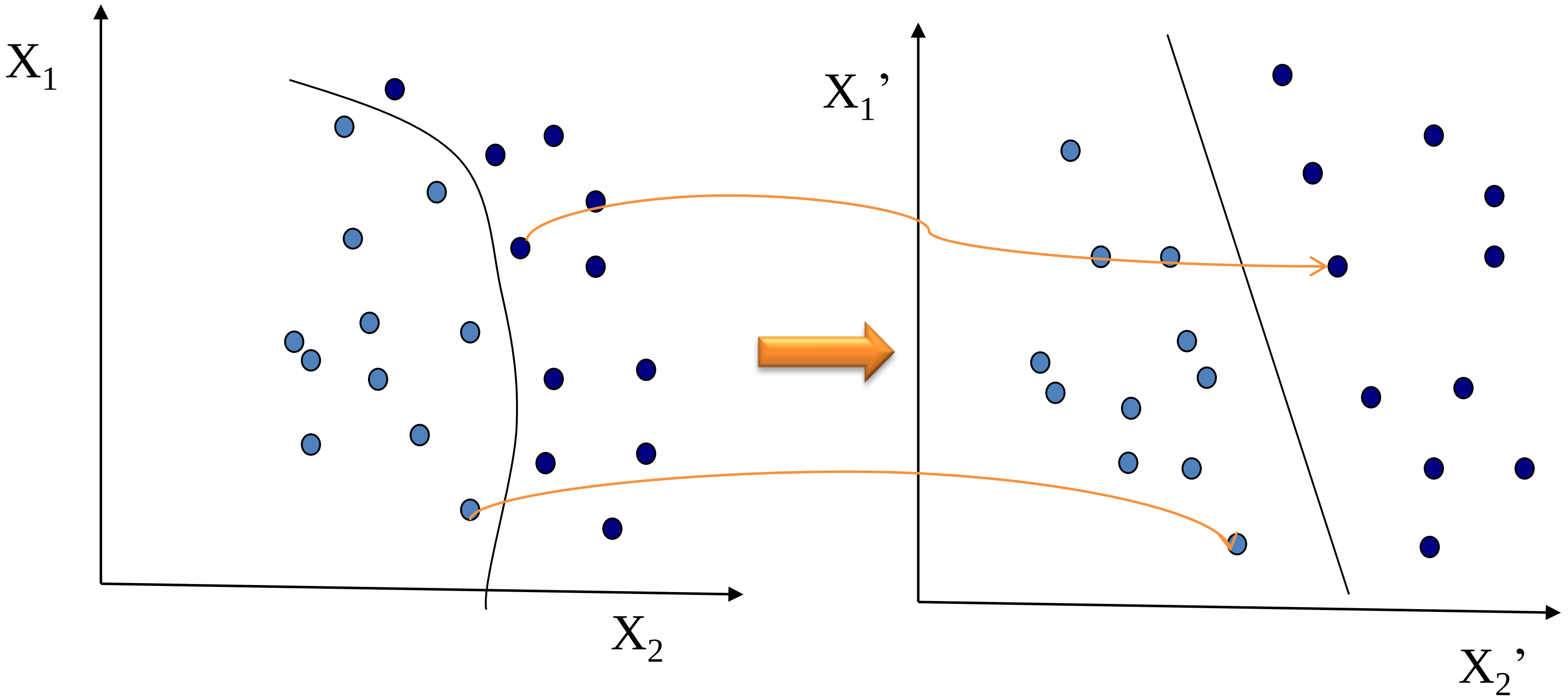
Soft vs Hard Margin SVM

- Soft-Margin always have a solution
- Soft-Margin is more robust to outliers
  - Smoother surfaces (in the non-linear case)
- Hard-Margin does not require to guess the cost parameter (requires no parameters at all)

Non-linear SVM

- The original optimal hyperplane algorithm proposed by Vladimir Vapnik in 1963 was a linear classifier.
- However, in 1992, Bernhard Boser, Isabelle Guyon and Vapnik suggested a way to create non-linear classifiers by applying the kernel trick to maximum-margin hyperplanes.
- The resulting algorithm is formally similar, except that every dot product is replaced by a non-linear kernel function.
- This allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space.
- The transformation may be non-linear and the transformed space high dimensionalthus though the classifier is a hyperplane in the high-dimensional feature space, it may be non-linear in the original input space.

Linear Classifiers in High-Dimensional Spaces



Find function $\Phi(x)$ to map to a different space

Mapping Data to a High-Dimensional Space

- Find function **Φ(x)** to map to a different space, then SVM formulation becomes:

$$\min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \qquad s.t.\ \ y_i(w \cdot \Phi(x) + b) \geq 1 - \xi_i, \forall x_i$$
$$\xi_i \geq 0$$

- Data appear as Φ(x), weights *w* are now weights in the new space.
- Explicit mapping expensive if Φ(x) is very high dimensional.
- Solving the problem without explicitly mapping the data is desirable.

The Kernel Trick

Linear SVM

$$x_i \cdot x_j$$

Non-linear SVM

$$\phi(x_i) \cdot \phi(x_j)$$

map data into new space, then take the inner product of the new vectors.

Kernel function

$$k(x_i \cdot x_j)$$

the image of the inner product of the data is the inner product of the images of the data.

SVM – Kernel functions

Polynomial

$$k(\mathrm{x}_i, \mathrm{x}_j) = (\mathrm{x}_i . \mathrm{x}_j)^d$$

Gaussian Radial Basis function

$$k(\mathrm{x}_i, \mathrm{x}_j) = \exp\left(-\frac{\left\| \mathrm{x}_i - \mathrm{x}_j \right\|^2}{2\sigma^2} \right)$$

Other Types of SVM

- SVMs that perform regression (SVR).
- SVMs that perform clustering.
- SVM formulations that take into consideration difference in cost of misclassification for the different classes.
- Kernels suitable for sequences of strings, or other specialized kernels.

Variable Selection with SVMs

- **Recursive Feature Elimination**
  - Train a linear/non-linear SVM.
  - Remove the variables with the lowest weights (those variables affect classification the least), e.g., remove the lowest 50% of variables.
  - Retrain the SVM with remaining variables and repeat until classification is reduced.
- Some of the best and most efficient variable selection methods.

MultiClass SVMs

- One-versus-all
  - Train *n* binary classifiers, one for each class against all other classes.
  - Predicted class is the class of the most confident classifier
- Truly MultiClass SVMs
  - Generalize the SVM formulation to multiple categories

Comparison with Neural Networks

Neural Networks

- Hidden Layers map to lower dimensional spaces
- Search space has multiple local minima
- Training is expensive
- Classification extremely efficient
- Requires number of hidden units and layers
- Very good accuracy in typical domains

SVMs

- Kernel maps to a very-high dimensional space
- Search space has a unique minimum
- Training is extremely efficient
- Classification extremely efficient
- Kernel and cost the two parameters to select
- Very good accuracy in typical domains
- Extremely robust

References

- www.dsl-lab.org/ml_tutorial/Presentation/file4.ppt
- http://en.wikipedia.org/wiki/Support_vector_machine