Support Vector machines

Supervised Learning Model

Topics Outline

- Linear Classifiers and Generalization
- Maximum Margin Classification
- Learning a Maximum Margin Classifier: SVM
- Non-Linear Feature Mapping
- The Kernel Trick

Linear Classifier

- It has linear partition or decision boundaries.
- Decision Boundary:

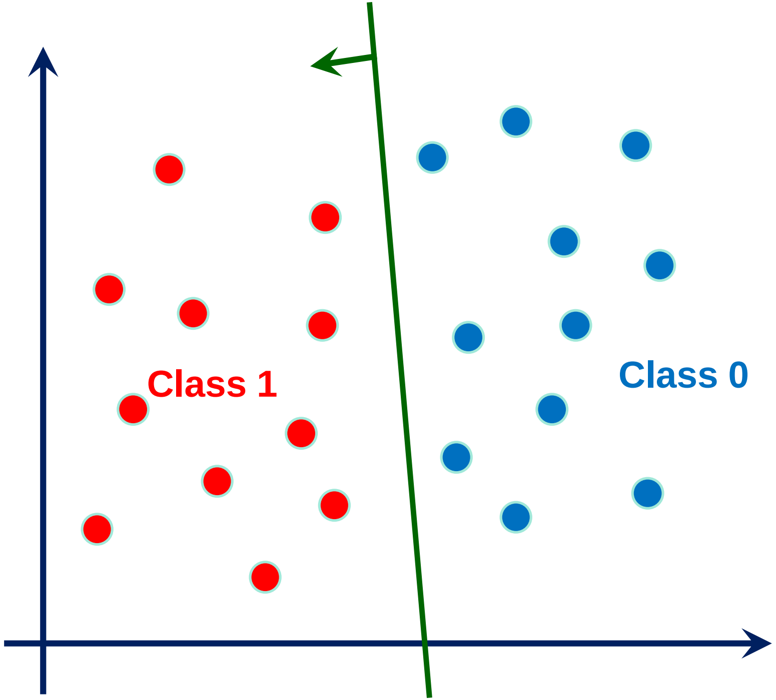
$$W^T X = 0$$

- Class 1 lies on the positive side

$$W^T X > 0$$

- Class 0 lies on the negative side

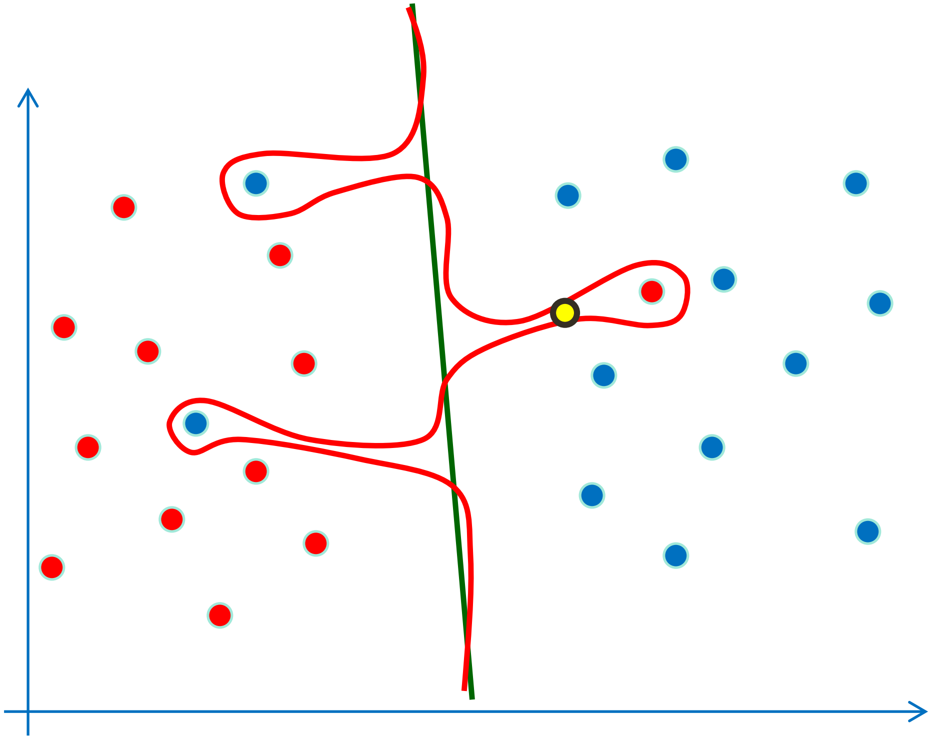
$$W^T X < 0$$



Why Linear? Generalization vs. Complexity

- Is it good to use a complex curve to reduce training error?

Are both solutions equally good?



Summary

- Linear Classifiers are simple and hence efficient
- There exists simple learning algorithms
- Likely to work well for unseen test data (generalization)
- Can be converted to non-linear ones (later)

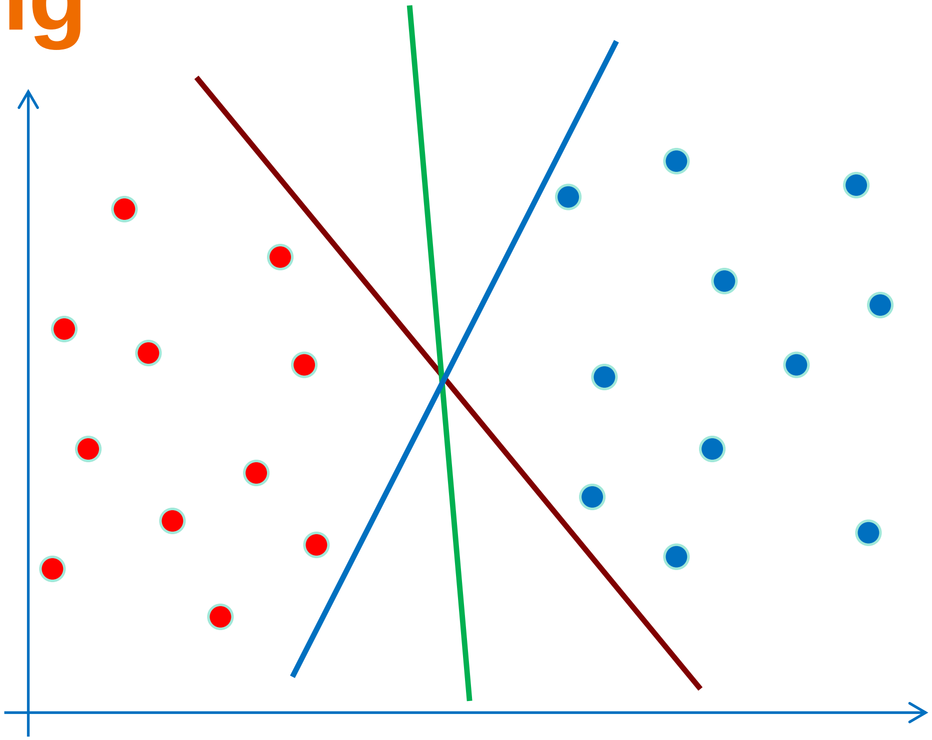
Topics Outline

- Linear Classifiers and Generalization
- Maximum Margin Classification
- Learning a Maximum Margin Classifier: SVM
- Non-Linear Feature Mapping
- The Kernel Trick

Perceptron Learning

- Multiple solutions exist for linearly separable data
- Perceptron learning (any GD) results in a feasible solution

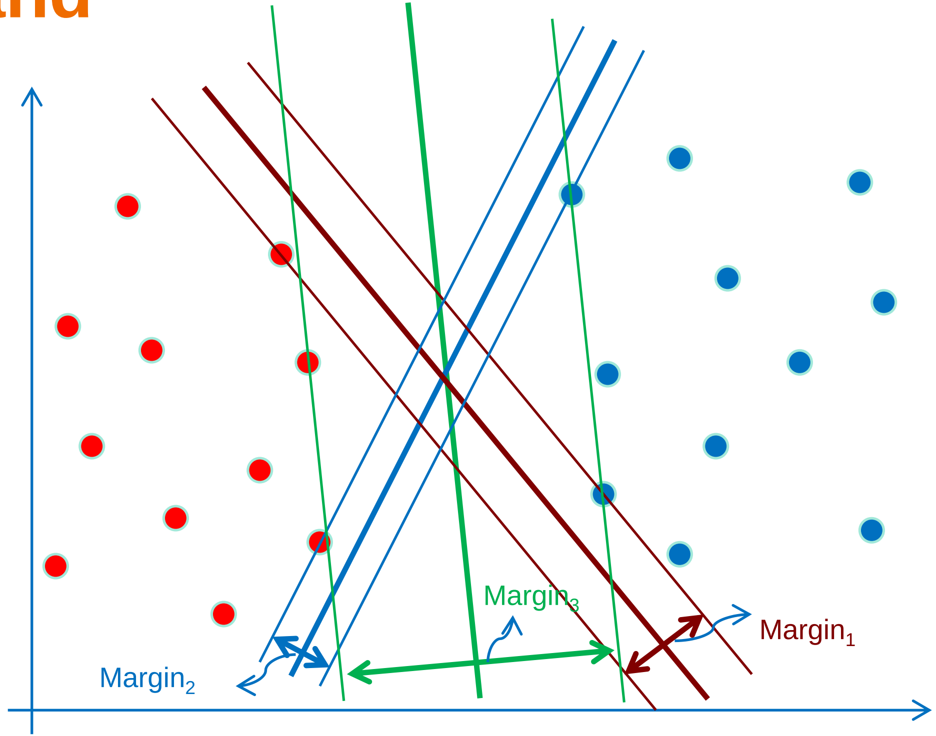
Are all solutions equally good?



Margin: The No-mans Band

- Margin: Width of a band around decision boundary without any training samples
- Margin varies with the position and orientation of the sep ane

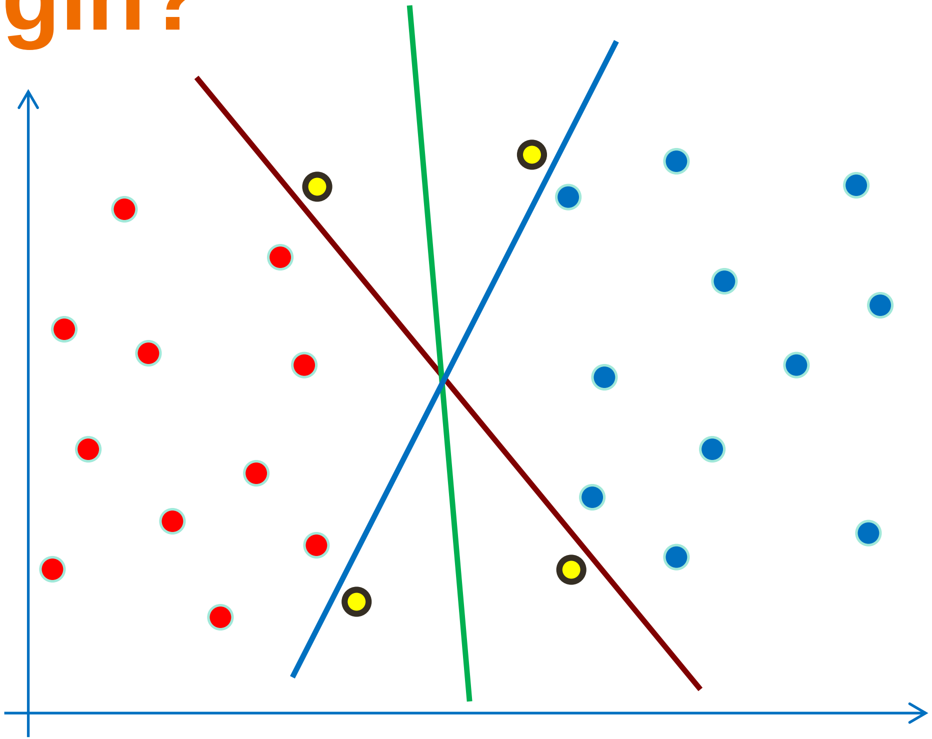
Is a Larger Margin better? Why?



Why Maximize Margin?

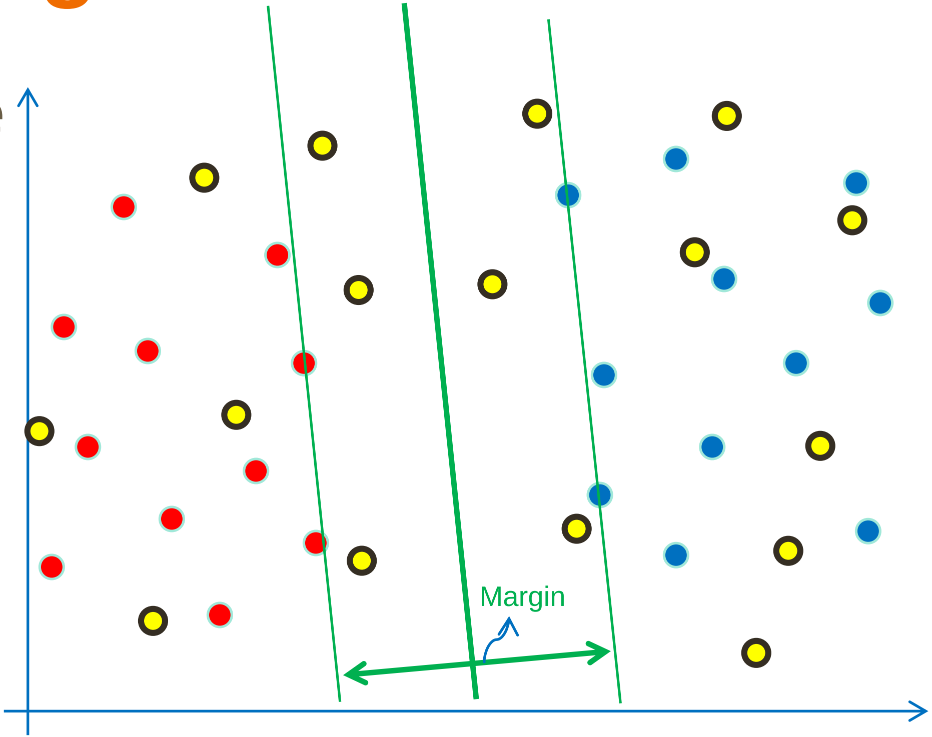
- Test samples vary from training data
- What is their chance of being misclassified?

Training and Test Samples come from the same population



Summary: Max-Margin Classification

- A Large Margin will reduce the chance of misclassifying future test samples
- In other words, large-margin classifiers will generalize better.



Topics Outline

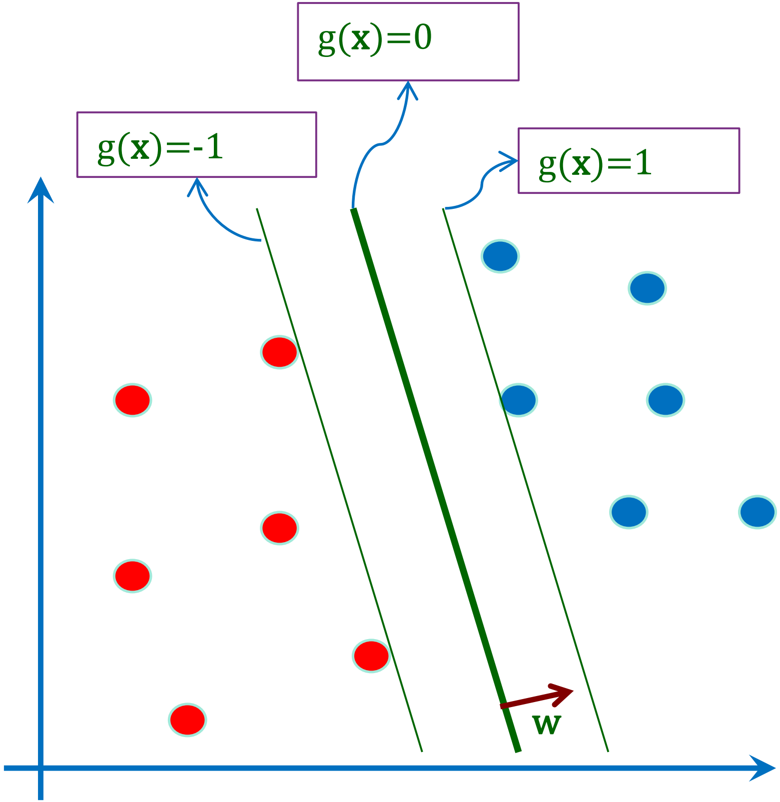
- Linear Classifiers and Generalization
- Maximum Margin Classification
- Learning a Maximum Margin Classifier: SVM
- Non-Linear Feature Mapping
- The Kernel Trick

SVM: Formulation

Let $g(X) = W^T X + b$

We want to maximize margin:

- $W^T X_i + b \leq -1$ for $y_i = -1$
- $W^T X_i + b \geq 1$ for $y_i = 1$
- Or $y_i(W^T X_i + b \geq 1)$ for all



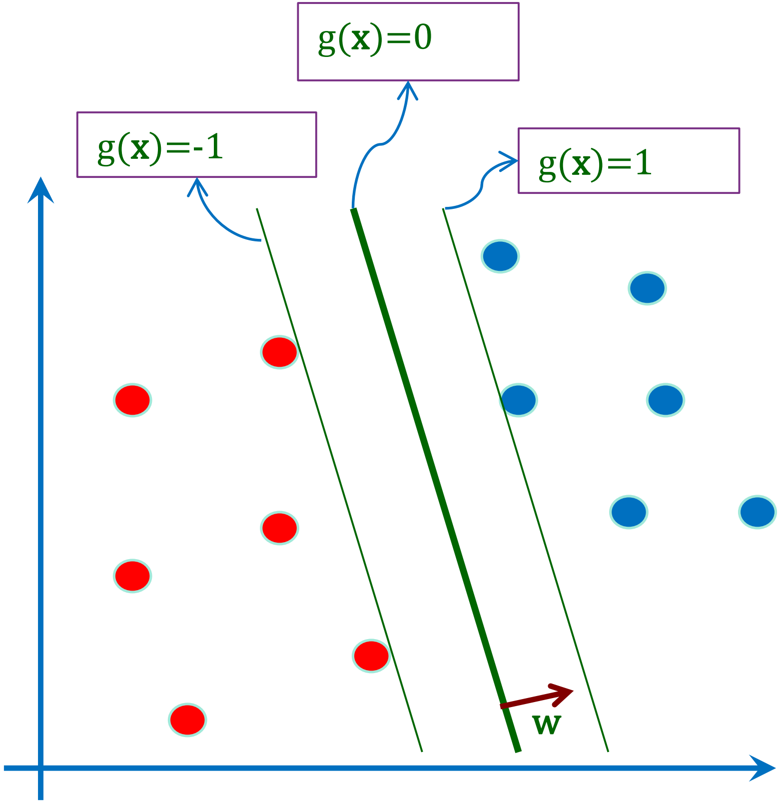
SVM: Formulation

- Mathematically,

  Maximize $\frac{1}{2}W^TW$

- Subject to:
  - $y_i(W^TX_i + b) \geq 1$ for all i.

- This is convex optimization. Exact solutions exist



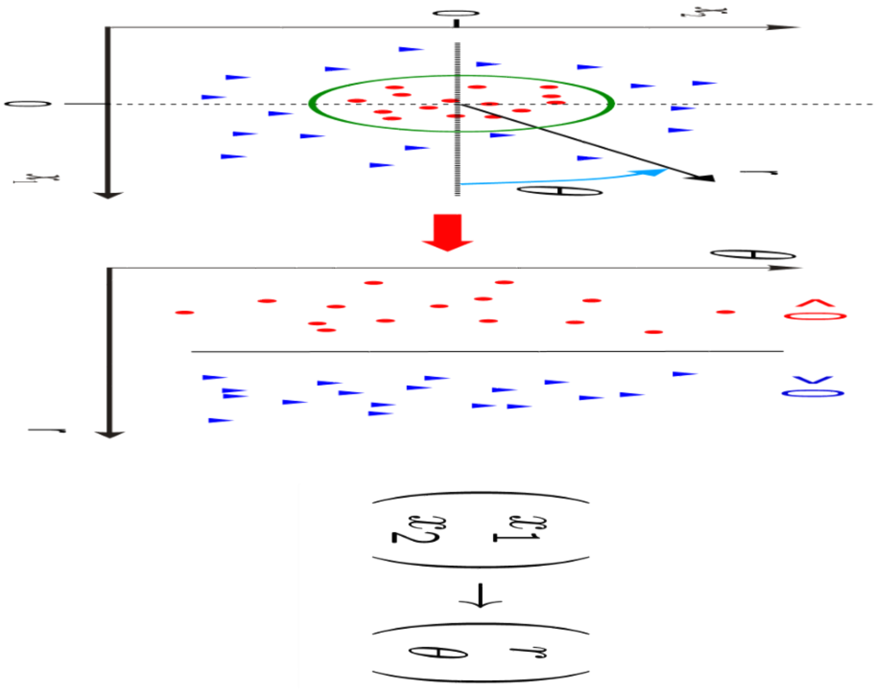
Summary

- SVMs are very good at generalization
- Convex optimization. No worries about local minima.
- Many excellent solvers. (Often we never code ourselves.)
- Linear SVMs are efficient for training and testing (both memory and flops).
- They are also highly accurate

Topics Outline

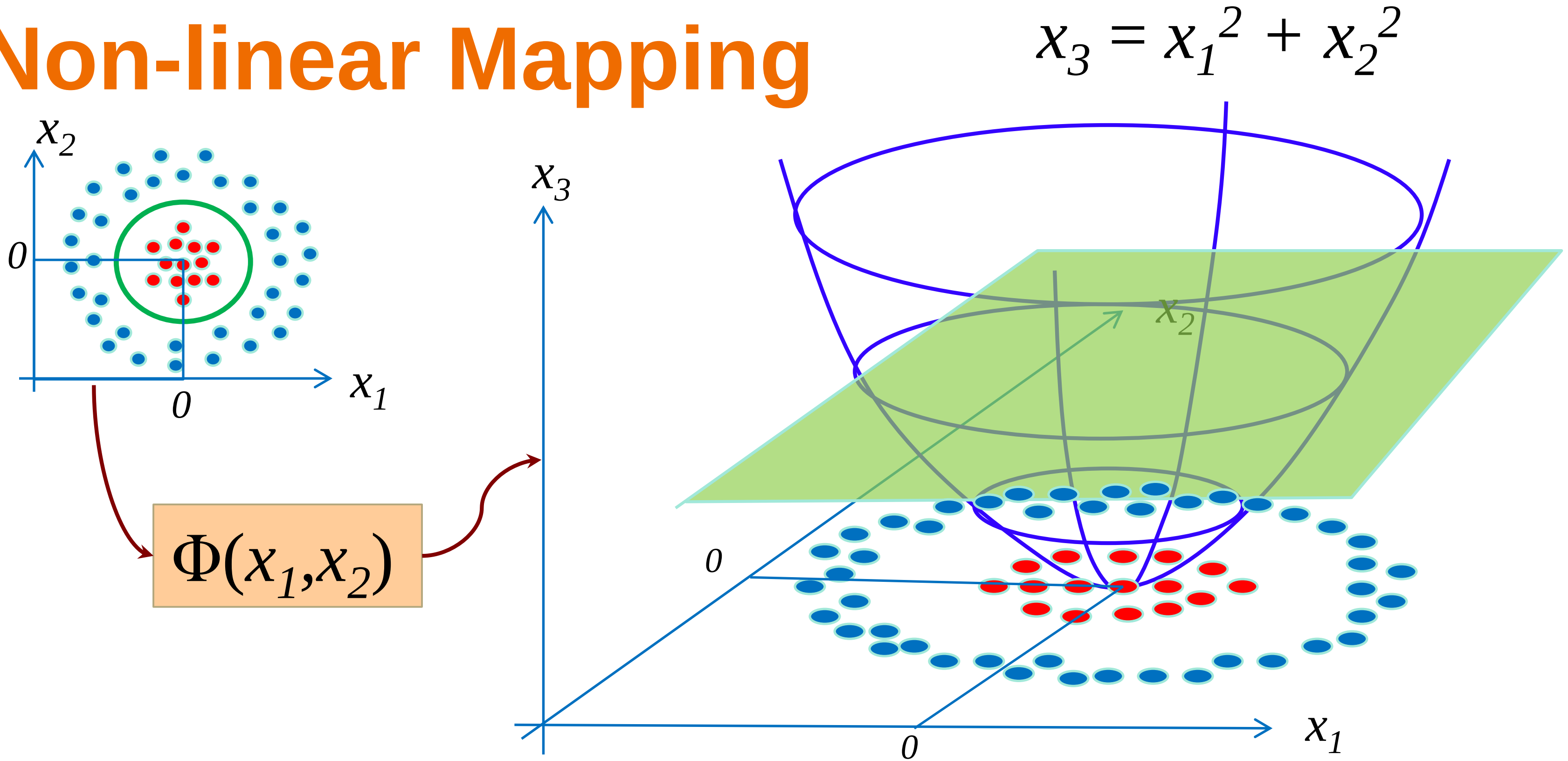
- Linear Classifiers and Generalization
- Maximum Margin Classification
- Learning a Maximum Margin Classifier: SVM
- Non-Linear Feature Mapping
- The Kernel Trick

Nonlinearity with Feature Maps



With a “smart” feature map, a linearly non-separable problem can be converted to a separable problem.!! The feature mapping is often denoted by: $\phi(X)$

Non-linear Mapping



Φ is a non-linear mapping into a possibly high-dimensional space

kernels

Similarity Function

Kernels

- Interestingly, it is possible to do this without explicitly doing the non-linear mapping to high dimensions
- We need only a kernel function $K(x_i, x_j)$

$$K(s_i, x_i) = \phi(s_i).\phi(x_i)$$

Popular Kernels

- Polynomial:

$$\boxed{\mathrm{K}_p(\mathbf{X},\mathbf{Y}) = (1 + \mathbf{X} \bullet \mathbf{Y})^p}$$

- Radial Basis Function (RBF) or Gaussian:

$$\boxed{\mathrm{K}_r(\mathbf{X},\mathbf{Y}) = e^{-\frac{1}{2\sigma^2}\|\mathbf{X}-\mathbf{Y}\|_2^2}}$$

- Hyperbolic Tangent:

$$\boxed{\mathrm{K}_s(\mathbf{X},\mathbf{Y}) = \tanh(\beta_0 \mathbf{X} \bullet \mathbf{Y} + \beta_1)}$$

Summary

- Linear SVMs generalize well, but cannot separate non-linear data
- Kernels (nonlinear) SVMs are also good at generalization, and can deal with non-linear data.
- Need not be as efficient/compact.

Thanks!

Questions?

SVM: Primal and Dual

$$min 1/2 W^T W$$

subject to $y_i(W^T x_i + b) - 1 \geq 0 \forall i$

This results in $y_i \epsilon \{1, -1\}$

maximization of

$$J_d(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \alpha_i \alpha j y_i y_j x_i^T x_j$$

$$W = \sum_{i=1}^{N} \alpha_i y_i x_i$$