**CS840a**
**Learning and Computer Vision**
**Prof. Olga Veksler**

# Lecture 4

SVM

Some pictures from C. Burges

## SVM

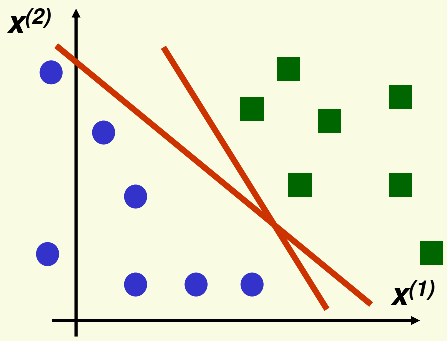
- Said to start in 1979 with Vladimir Vapnik's paper
- Major developments throughout 1990's
- Elegant theory
  - Has good generalization properties
- Have been applied to diverse problems very successfully in the last 10-15 years
- One of the most important developments in pattern recognition in the last 10 years

## Linear Discriminant Functions

- A discriminant function is linear if it can be written as

$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0$$

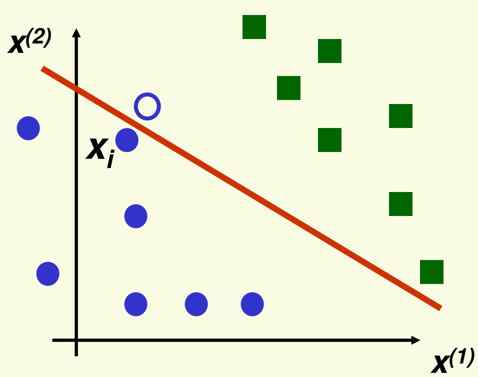
$$\boxed{\begin{array}{l} g(\mathbf{x}) > 0 \quad \Rightarrow \mathbf{x} \in \text{class } 1 \\ g(\mathbf{x}) < 0 \quad \Rightarrow \mathbf{x} \in \text{class } 2 \end{array}}$$




- which separating hyperplane should we choose?

## Linear Discriminant Functions

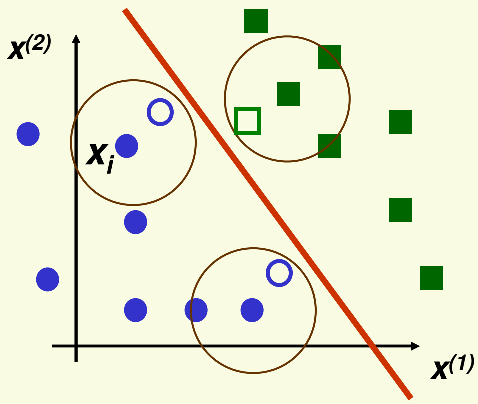
- Training data is just a subset of of all possible data
- Suppose hyperplane is close to sample $\mathbf{x}_i$
- If we see new sample close to sample $i$, it is likely to be on the wrong side of the hyperplane




- Poor generalization (performance on unseen data)

## Linear Discriminant Functions

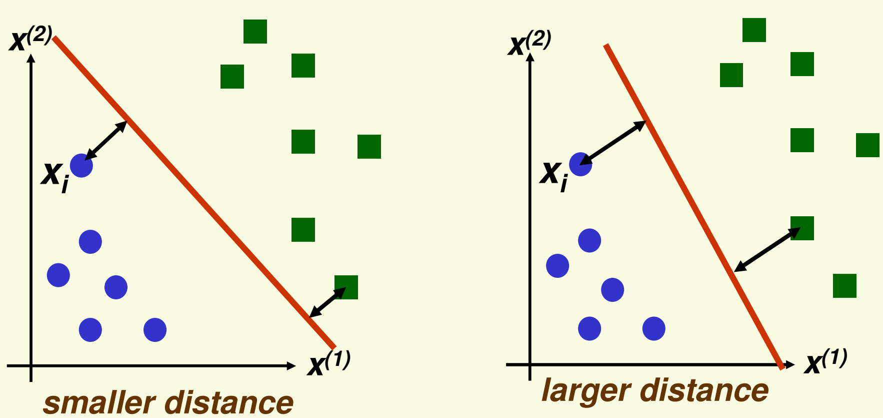
- Hyperplane as far as possible from any sample




- New samples close to the old samples will be classified correctly
- Good generalization

## SVM

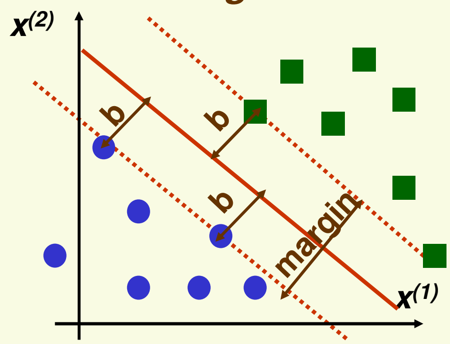
- Idea: maximize distance to the closest example




- For the optimal hyperplane
  - distance to the closest negative example = distance to the closest positive example

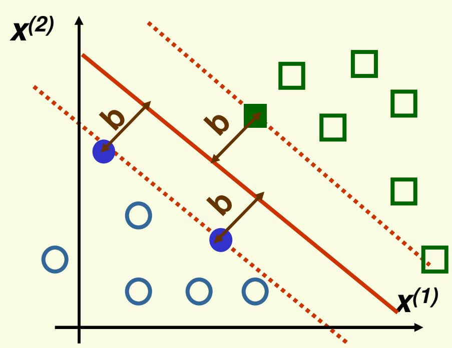
## SVM: Linearly Separable Case

- SVM: maximize the ***margin***



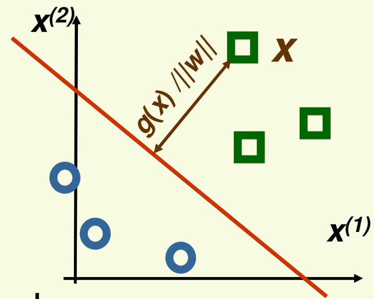

- ***margin*** is twice the absolute value of distance ***b*** of the closest example to the separating hyperplane
- Better generalization (performance on test data)
  - in practice
  - and in theory

## SVM: Linearly Separable Case




- ***Support vectors*** are the samples closest to the separating hyperplane
  - they are the most difficalt patterns to classify
  - Optimal hyperplane is completely defined by support vectors
    - of course, we do not know which samples are support vectors without finding the optimal hyperplane

## SVM: Formula for the Margin




- $g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0$
- absolute distance between $\mathbf{x}$ and the boundary $g(\mathbf{x}) = 0$

$$\frac{|\mathbf{w}^t \mathbf{x} + w_0|}{\|\mathbf{w}\|}$$

- distance is unchanged for hyperplane $g_1(\mathbf{x}) = \alpha g(\mathbf{x})$

$$\frac{|\alpha \mathbf{w}^t \mathbf{x} + \alpha w_0|}{\|\alpha \mathbf{w}\|} = \frac{|\mathbf{w}^t \mathbf{x} + w_0|}{\|\mathbf{w}\|}$$

- Let $\mathbf{x}_i$ be an example closest to the boundary. Set

$$|\mathbf{w}^t \mathbf{x}_i + w_0| = 1$$

- Now the largest margin hyperplane is unique

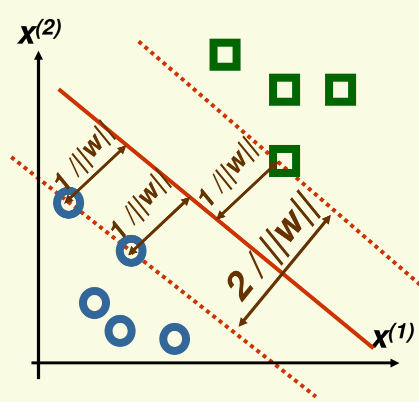
## SVM: Formula for the Margin

- For uniqueness, set $|\mathbf{w}^t \mathbf{x}_i + w_0| = 1$ for any example $\mathbf{x}_i$ closest to the boundary
- now distance from closest sample $\mathbf{x}_i$ to $g(\mathbf{x}) = 0$ is

$$\frac{|\mathbf{w}^t \mathbf{x}_i + w_0|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$$

- Thus the margin is

$$\boxed{m = \frac{2}{\|\mathbf{w}\|}}$$



## SVM: Optimal Hyperplane

- Maximize margin $m = \frac{2}{\|w\|}$
  - subject to constraints

$$\begin{cases} w^t x_i + w_0 \geq 1 & \text{if } x_i \text{ is positive example} \\ w^t x_i + w_0 \leq -1 & \text{if } x_i \text{ is negative example} \end{cases}$$

- Let $\begin{cases} z_i = 1 & \text{if } x_i \text{ is positive example} \\ z_i = -1 & \text{if } x_i \text{ is negative example} \end{cases}$
- Can convert our problem to

$$\boxed{\begin{array}{ll} \text{minimize} & J(w) = \frac{1}{2}\|w\|^2 \\ \text{constrained to} & z_i\left(w^t x_i + w_0\right) \geq 1 \quad \forall i \end{array}}$$

- $J(w)$ is a quadratic function, thus there is a single global minimum

## SVM: Optimal Hyperplane

- Use Kuhn-Tucker theorem to convert our problem to:

$$\boxed{\begin{array}{ll} \text{maximize} & L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j \\ \text{constrained to} & \alpha_i \geq 0 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0 \end{array}}$$

- $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ are new variables, one for each sample
- Can rewrite $L_D(\alpha)$ using $n$ by $n$ matrix $H$:

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}^t H \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}$$

  - where the value in the $i$th row and $j$th column of $H$ is

$$H_{ij} = z_i z_j x_i^t x_j$$

## SVM: Optimal Hyperplane

- Use Kuhn-Tucker theorem to convert our problem to:

$$\boxed{\begin{array}{ll} \text{maximize} & L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j \\ \text{constrained to} & \alpha_i \geq 0 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0 \end{array}}$$

- $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ are new variables, one for each sample
- $L_D(\alpha)$ can be optimized by quadratic programming
- $L_D(\alpha)$ formulated in terms of $\alpha$
  - it depends on $w$ and $w_0$ indirectly

## SVM: Optimal Hyperplane

- After finding the optimal $\alpha = \{\alpha_1, \ldots, \alpha_n\}$
  - For every sample $i$, one of the following must hold
    - $\alpha_i = 0$ (sample $i$ is not a support vector)
    - $\alpha_i \neq 0$ and $z_i(w^t x_i + w_0 - 1) = 0$ (sample $i$ is support vector)
  - can find $w$ using $w = \sum_{i=1}^{n} \alpha_i z_i x_i$
  - can solve for $w_0$ using any $\alpha_i > 0$ and $\alpha_i\left[z_i\left(w^t x_i + w_0\right) - 1\right] = 0$

$$w_0 = \frac{1}{z_i} - w^t x_i$$

- Final discriminant function:

$$g(x) = \left(\sum_{x_i \in S} \alpha_i z_i x_i\right)^t x + w_0$$

  - where $S$ is the set of support vectors

$$S = \{x_i \mid \alpha_i \neq 0\}$$

## SVM: Optimal Hyperplane

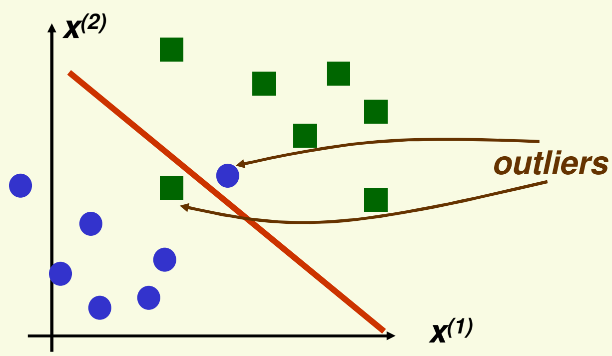
$$\text{maximize} \quad L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j$$

$$\text{constrained to} \quad \alpha_i \geq 0 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0$$

- $L_D(\alpha)$ depends on the number of samples, not on dimension of samples
- samples appear only through the dot products $x_i^t x_j$
- This will become important when looking for a ***nonlinear*** discriminant function, as we will see soon
- Code available on the web to optimize

## SVM: Non Separable Case

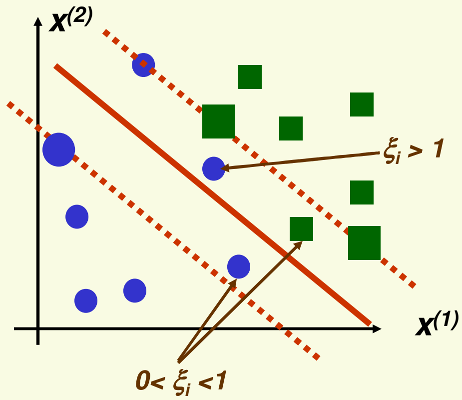
- Data is most likely to be not linearly separable, but linear classifier may still be appropriate




- Can apply SVM in non linearly separable case
  - data should be "almost" linearly separable for good performance

## SVM: Non Separable Case

- Use non-negative slack variables $\xi_1, \ldots, \xi_n$ (one for each sample)
- Change constraints from $z_i(w^t x_i + w_0) \geq 1 \quad \forall i$ to
  $$z_i(w^t x_i + w_0) \geq 1 - \xi_i \quad \forall i$$
- $\xi_i$ is a measure of deviation from the ideal for sample ***i***
  - $\xi_i > 1$ sample ***i*** is on the wrong side of the separating hyperplane
  - $0 < \xi_i < 1$ sample ***i*** is on the right side of separating hyperplane but within the region of maximum margin




## SVM: Non Separable Case

- Would like to minimize

$$J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2}\|w\|^2 + \beta \sum_{i=1}^{n} I(\xi_i > 0)$$

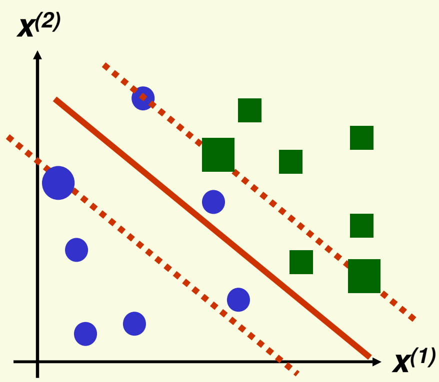
***# of samples not in ideal location***

  - where $I(\xi_i > 0) = \begin{cases} 1 & \text{if } \xi_i > 0 \\ 0 & \text{if } \xi_i \leq 0 \end{cases}$
  - constrained to $z_i(w^t x_i + w_0) \geq 1 - \xi_i$ and $\xi_i \geq 0 \quad \forall i$
  - $\beta$ is a constant which measures relative weight of the first and second terms
    - if $\beta$ is small, we allow a lot of samples not in ideal position
    - if $\beta$ is large, we want to have very few samples not in ideal positon

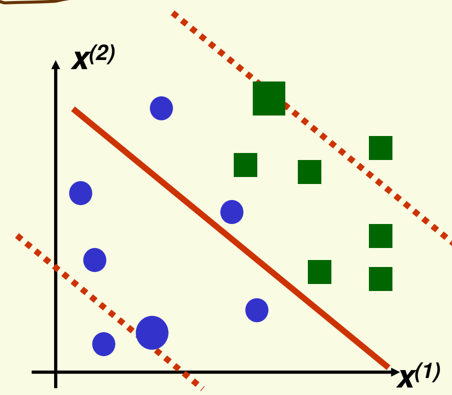
## SVM: Non Separable Case

$$J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2}\|w\|^2 + \beta \sum_{i=1}^{n} I(\xi_i > 0)$$

*# of examples not in ideal location*




*large $\beta$, few samples not in ideal position*




*small $\beta$, a lot of samples not in ideal position*

## SVM: Non Separable Case

- Unfortunately this minimization problem is NP-hard due to discontinuity of functions $I(\xi_i)$

$$J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2}\|w\|^2 + \beta \sum_{i=1}^{n} I(\xi_i > 0)$$

*# of examples not in ideal location*

  - where $I(\xi_i > 0) = \begin{cases} 1 & \text{if } \xi_i > 0 \\ 0 & \text{if } \xi_i \le 0 \end{cases}$
  - constrained to $z_i(w^t x_i + w_0) \ge 1 - \xi_i$ and $\xi_i \ge 0 \quad \forall i$

## SVM: Non Separable Case

- Instead we minimize

$$J(w,\xi_1,\ldots,\xi_n) = \frac{1}{2}\|w\|^2 + \beta\sum_{i=1}^{n}\xi_i$$

*a measure of # of misclassified examples*

  - constrained to $\begin{cases} z_i(w^t x_i + w_0) \ge 1 - \xi_i & \forall i \\ \xi_i \ge 0 & \forall i \end{cases}$

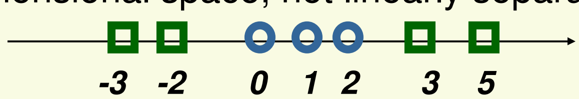
- Can use Kuhn-Tucker theorem to converted to

$$\boxed{\begin{array}{ll} \text{maximize} & L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j z_i z_j x_i^t x_j \\ \text{constrained to} & 0 \le \alpha_i \le \beta \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n}\alpha_i z_i = 0 \end{array}}$$

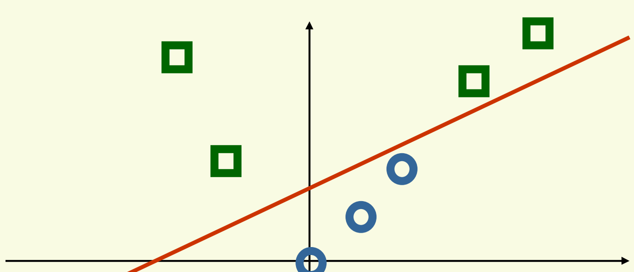
  - find $w$ using $w = \sum_{i=1}^{n}\alpha_i z_i x_i$
  - solve for $w_0$ using any $0 < \alpha_i < \beta$ and $\alpha_i\left[z_i\left(w^t x_i + w_0\right) - 1\right] = 0$

## Non Linear Mapping

- Cover's theorem:
  - "*pattern-classification problem cast in a high dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space*"
- One dimensional space, not linearly separable

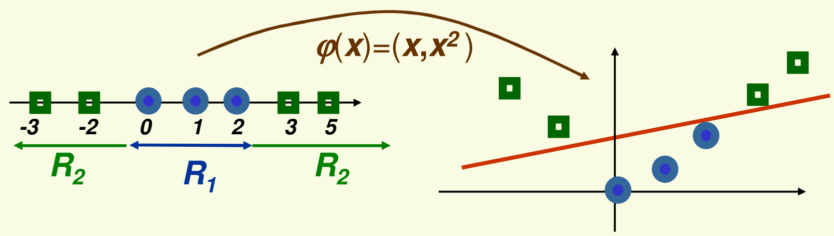



- Lift to two dimensional space with $\varphi(x) = (x, x^2)$



## Non Linear Mapping

- To solve a non linear classification problem with a linear classifier
  1. Project data $\boldsymbol{x}$ to high dimension using function $\varphi(\boldsymbol{x})$
  2. Find a linear discriminant function for transformed data $\varphi(\boldsymbol{x})$
  3. Final nonlinear discriminant function is $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{w}^t \varphi(\boldsymbol{x}) + \boldsymbol{w}_0$

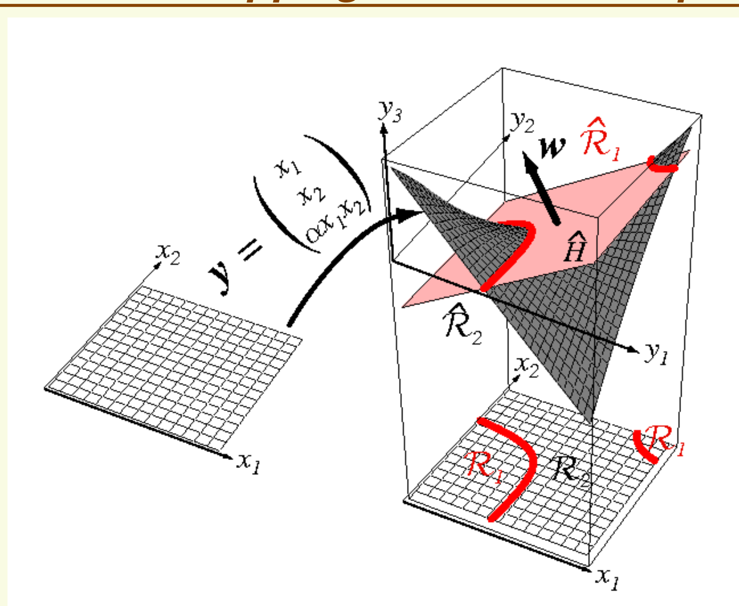



- In 2D, discriminant function is linear

$$g\left(\begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix}\right) = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix} + w_0$$

- In 1D, discriminant function is not linear $\quad g(x) = w_1 x + w_2 x^2 + w_0$

## Non Linear Mapping: Another Example



## *Non Linear SVM*

- Can use any linear classifier after lifting data into a higher dimensional space. However we will have to deal with the "curse of dimensionality"
  1. poor generalization to test data
  2. computationally expensive

- SVM avoids the "curse of dimensionality" problems by
  1. enforcing largest margin permits good generalization
     - It can be shown that generalization in SVM is a function of the margin, independent of the dimensionality
  2. computation in the higher dimensional case is performed only implicitly through the use of ***kernel*** functions

## *Non Linear SVM: Kernels*

- Recall SVM optimization

maximize $$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i z_i z_j x_i^t x_j$$

- Note this optimization depends on samples $\boldsymbol{x_i}$ only through the dot product $\boldsymbol{x_i^t x_j}$
- If we lift $\boldsymbol{x_i}$ to high dimension using $\varphi(\boldsymbol{x})$, need to compute high dimensional product $\varphi(\boldsymbol{x_i})^t \varphi(\boldsymbol{x_j})$

maximize $$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i z_i z_j \underbrace{\varphi(x_i)^t \varphi(x_j)}_{K(x_i,x_j)}$$

- Idea: find ***kernel*** function $\boldsymbol{K(x_i,x_j)}$ s.t.

$$\boldsymbol{K(x_i,x_j)} = \varphi(\boldsymbol{x_i})^t \varphi(\boldsymbol{x_j})$$

## Non Linear SVM: Kernels

maximize $L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j \underbrace{\varphi(x_i)^t \varphi(x_j)}_{K(x_i, x_j)}$

- Then we only need to compute $K(x_i, x_j)$ instead of $\varphi(x_i)^t \varphi(x_j)$
  - "kernel trick": do not need to perform operations in high dimensional space explicitly

## Non Linear SVM: Kernels

- Suppose we have 2 features and $K(x,y) = (x^t y)^2$
- Which mapping $\varphi(x)$ does it correspond to?

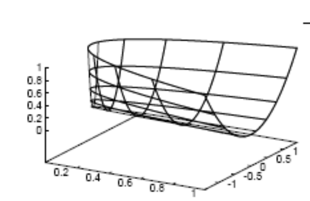
$$K(x,y) = (x^t y)^2 = \left( \begin{bmatrix} x^{(1)} & x^{(2)} \end{bmatrix} \begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix} \right)^2 = \left( x^{(1)} y^{(1)} + x^{(2)} y^{(2)} \right)^2$$

$$= \left( x^{(1)} y^{(1)} \right)^2 + 2\left( x^{(1)} y^{(1)} \right)\left( x^{(2)} y^{(2)} \right) + \left( x^{(2)} y^{(2)} \right)^2$$

$$= \begin{bmatrix} \left(x^{(1)}\right)^2 & \sqrt{2}\, x^{(1)} x^{(2)} & \left(x^{(2)}\right)^2 \end{bmatrix} \begin{bmatrix} \left(y^{(1)}\right)^2 & \sqrt{2}\, y^{(1)} y^{(2)} & \left(y^{(2)}\right)^2 \end{bmatrix}^t$$

- Thus

$\varphi(x) = \begin{bmatrix} \left(x^{(1)}\right)^2 & \sqrt{2}\, x^{(1)} x^{(2)} & \left(x^{(2)}\right)^2 \end{bmatrix}$



## Non Linear SVM: Kernels

- How to choose kernel function $\boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)$?
  - $\boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)$ should correspond to product $\varphi(\boldsymbol{x}_i)^t\varphi(\boldsymbol{x}_j)$ in a higher dimensional space
  - Mercer's condition tells us which kernel function can be expressed as dot product of two vectors
  - Kernel's not satisfying Mercer's condition can be sometimes used, but no geometrical interpretation
- Some common choices (satisfying Mercer's condition):
  - Polynomial kernel $K(x_i, x_j) = (x_i^t x_j + 1)^p$
  - Gaussian radial Basis kernel (data is lifted in infinite dimension)

$$K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2\right)$$

## Non Linear SVM

- search for separating hyperplane in high dimension

$$w\varphi(x) + w_0 = 0$$

- Choose $\varphi(\boldsymbol{x})$ so that the first ("0"th) dimension is the augmented dimension with feature value fixed to 1

$$\varphi(x) = \begin{bmatrix} 1 & x^{(1)} & x^{(2)} & x^{(1)}x^{(2)} \end{bmatrix}^t$$

- Threshold parameter $\boldsymbol{w_0}$ gets folded into the weight vector $\boldsymbol{w}$

$$\begin{bmatrix} w_0 & w \end{bmatrix} \underbrace{\begin{bmatrix} 1 \\ * \end{bmatrix}}_{\varphi(x)} = 0$$

## Non Linear SVM

- Will not use notation $\boldsymbol{a} = [\boldsymbol{w_0} \ \ \boldsymbol{w}]$, we'll use old notation $\boldsymbol{w}$ and seek hyperplane through the origin

$$w\varphi(x) = 0$$

- If the first component of $\varphi(\boldsymbol{x})$ is not $\boldsymbol{1}$, the above is equivalent to saying that the hyperplane has to go through the origin in high dimension
  - removes only one degree of freedom
  - But we have introduced many new degrees when we lifted the data in high dimension

## Non Linear SVM Recepie

- Start with data $\boldsymbol{x_1},\ldots,\boldsymbol{x_n}$ which lives in feature space of dimension $\boldsymbol{d}$
- Choose kernel $\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{x_j})$ or function $\varphi(\boldsymbol{x_i})$ which takes sample $\boldsymbol{x_i}$ to a higher dimensional space
- Find the largest margin linear discriminant function in the higher dimensional space by using quadratic programming package to solve:

$$\text{maximize} \quad L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j K(x_i, x_j)$$

$$\text{constrained to} \quad 0 \le \alpha_i \le \beta \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0$$

## Non Linear SVM Recipe

- Weight vector $\boldsymbol{w}$ in the high dimensional space:

$$w = \sum_{x_i \in S} \alpha_i z_i \varphi(x_i)$$

  - where $\boldsymbol{S}$ is the set of support vectors $S = \{x_i \mid \alpha_i \neq 0\}$

- Linear discriminant function of largest margin in the high dimensional space:

$$g(\varphi(x)) = w^t \varphi(x) = \left(\sum_{x_i \in S} \alpha_i z_i \varphi(x_i)\right)^t \varphi(x)$$

- Non linear discriminant function in the original space

$$g(x) = \left(\sum_{x_i \in S} \alpha_i z_i \varphi(x_i)\right)^t \varphi(x) = \sum_{x_i \in S} \alpha_i z_i \varphi^t(x_i)\varphi(x) = \sum_{x_i \in S} \alpha_i z_i K(x_i, x)$$

- decide class 1 if $\boldsymbol{g}(\boldsymbol{x}) > 0$, otherwise decide class 2

## Non Linear SVM

- Nonlinear discriminant function

$$g(x) = \sum_{x_i \in S} \boxed{\alpha_i} \boxed{z_i} \boxed{K(x_i, x)}$$

$$g(x) = \sum$$

most important training samples, i.e. support vectors

weight of support vector $\boldsymbol{x_i}$

$\mp 1$

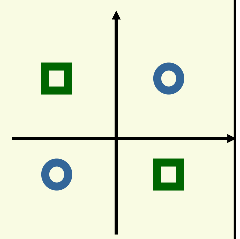
"inverse distance" from $\boldsymbol{x}$ to support vector $\boldsymbol{x_i}$

$$K(x_i, x) = \exp\left(-\frac{1}{2\sigma^2}\|x_i - x\|^2\right)$$

## SVM Example: XOR Problem

- Class 1: $\mathbf{x_1} = [1,-1]$, $\mathbf{x_2} = [-1,1]$
- Class 2: $\mathbf{x_3} = [1,1]$, $\mathbf{x_4} = [-1,-1]$
- Use polynomial kernel of degree 2:
  - $K(\mathbf{x}_i,\mathbf{x}_j) = (\mathbf{x}_i^{\,t}\mathbf{x}_j + 1)^2$
  - This kernel corresponds to mapping

$$\varphi(x) = \left[1 \quad \sqrt{2}x^{(1)} \quad \sqrt{2}x^{(2)} \quad \sqrt{2}x^{(1)}x^{(2)} \quad \left(x^{(1)}\right)^2 \quad \left(x^{(2)}\right)^2\right]^t$$



- Need to maximize

$$L_D(\alpha) = \sum_{i=1}^{4}\alpha_i - \frac{1}{2}\sum_{i=1}^{4}\sum_{j=1}^{4}\alpha_i\alpha_j z_i z_j\left(x_i^t x_j + 1\right)^2$$

constrained to $0 \le \alpha_i \quad \forall i \quad and \quad \alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0$

## SVM Example: XOR Problem

- Can rewrite $L_D(\alpha) = \sum_{i=1}^{4}\alpha_i - \frac{1}{2}\alpha^t H\alpha$
  - where $\alpha = [\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4]^t$ and $H = \begin{bmatrix} 9 & 1 & -1 & -1 \\ 1 & 9 & -1 & -1 \\ -1 & -1 & 9 & 1 \\ -1 & -1 & 1 & 9 \end{bmatrix}$
- Take derivative with respect to $\alpha$ and set it to $\boldsymbol{0}$

$$\frac{d}{da}L_D(\alpha) = \begin{bmatrix}1\\1\\1\\1\end{bmatrix} - \begin{bmatrix} 9 & 1 & -1 & -1 \\ 1 & 9 & -1 & -1 \\ -1 & -1 & 9 & 1 \\ -1 & -1 & 1 & 9 \end{bmatrix}\alpha = 0$$

- Solution to the above is $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \mathbf{0.25}$
  - satisfies the constraints $\forall i, \ 0 \le \alpha_i \ and \ \alpha_1 + \alpha_2 - \alpha_3 - \alpha_4 = 0$
  - all samples are support vectors

## SVM Example: XOR Problem

$$\varphi(x) = \begin{bmatrix} 1 & \sqrt{2}x^{(1)} & \sqrt{2}x^{(2)} & \sqrt{2}x^{(1)}x^{(2)} & (x^{(1)})^2 & (x^{(2)})^2 \end{bmatrix}^t$$

- Weight vector $w$ is:

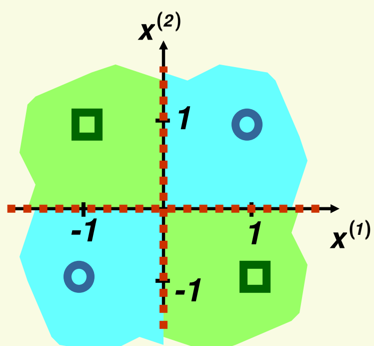
$$w = \sum_{i=1}^{4} \alpha_i z_i \varphi(x_i) = 0.25(\varphi(x_1) + \varphi(x_2) - \varphi(x_3) - \varphi(x_4))$$
$$= \begin{bmatrix} 0 & 0 & 0 & -\sqrt{2} & 0 & 0 \end{bmatrix}$$

- Thus the nonlinear discriminant function is:

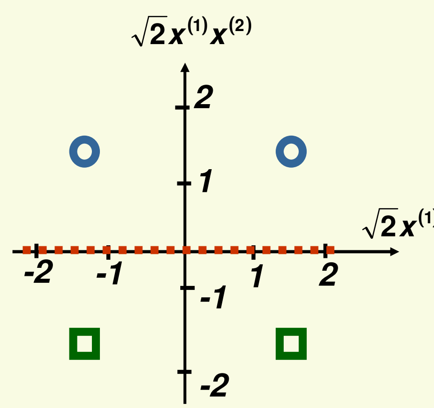
$$g(x) = w\varphi(x) = \sum_{i=1}^{6} w_i \varphi_i(x) = -\sqrt{2}\left(\sqrt{2}x^{(1)}x^{(2)}\right) = -2x^{(1)}x^{(2)}$$

## SVM Example: XOR Problem

$$g(x) = -2x^{(1)}x^{(2)}$$

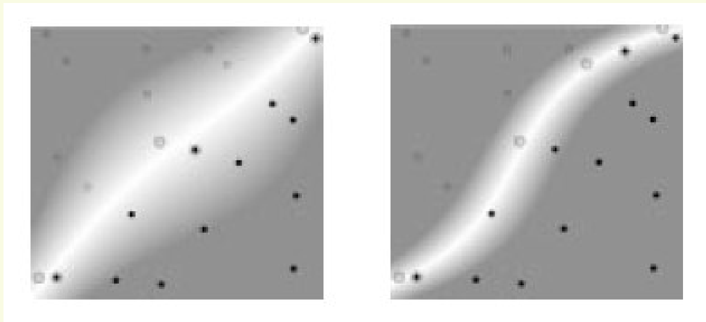



decision boundaries nonlinear




decision boundary is linear

## Degree 3 Polynomial Kernel



- In linearly separable case (on the left), decision boundary is roughly linear, indicating that dimensionality is controlled
- Nonseparable case (on the right) is handled by a polynomial of degree 3

## SVM Summary

- Advantages:
  - Based on nice theory
  - excellent generalization properties
  - objective function has no local minima
  - can be used to find non linear discriminant functions
  - Complexity of the classifier is characterized by the number of support vectors rather than the dimensionality of the transformed space
- Disadvantages:
  - tends to be slower than other methods
  - quadratic programming is computationally expensive
  - Not clear how to choose the Kernel