

# CS 66: Machine Learning

Prof. Sara Mathieson

Spring 2019



# Outline for March 27

- Finish Lagrangian for SVMs
- Kernels
- Soft-margin SVMs

## Admin

- \* last call for partners for Lab 6
- \* final project presentations

- Thurs May 16

9-12 pm

- (tentatively)

Mon May 13

4-6 pm

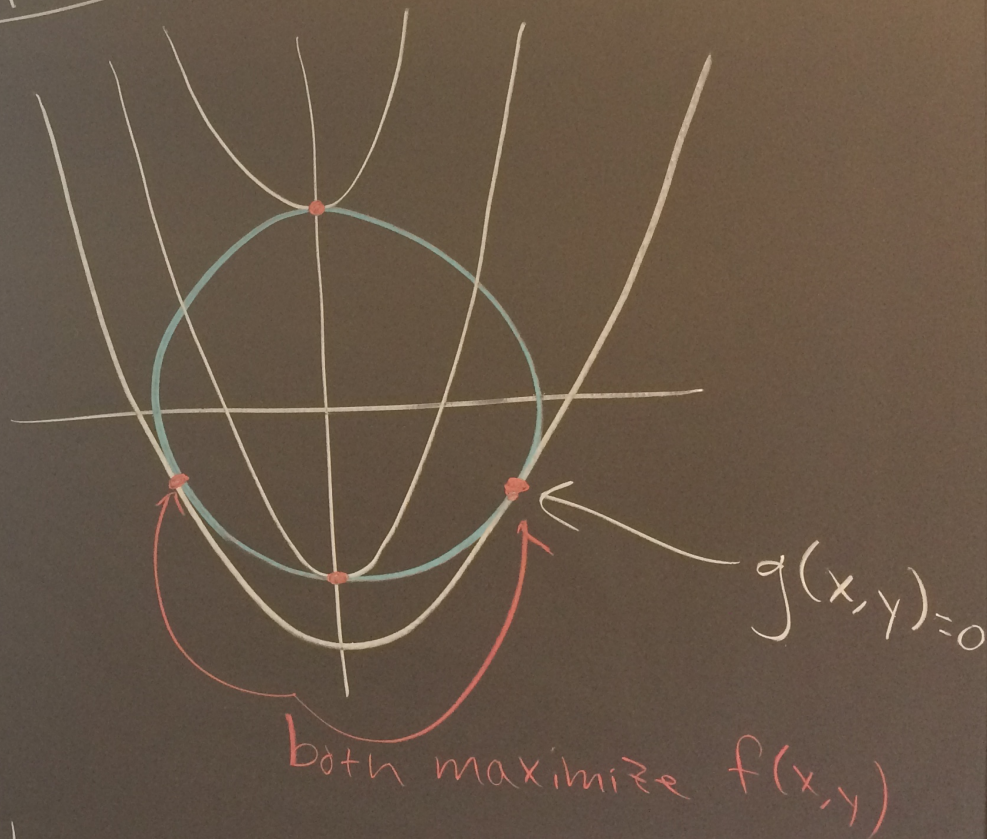
\* Lab 3 back Friday!

\* Lab 6: SVMs

\* Lab 7: Deep Learning

} both 1.5 weeks

## Handout 11, Q2



# Outline for March 27

- Finish Lagrangian for SVMs
- Kernels
- Soft-margin SVMs

## SVM Goal

$$\begin{aligned} \min_{\vec{w}, b} \quad & \frac{1}{2} \|\vec{w}\|^2 \\ \text{s.t.} \quad & -y_i(\vec{w} \cdot \vec{x}_i + b) + 1 \leq 0 \\ & i = 1, 2, \dots, n \end{aligned}$$

$\rightarrow \hat{\gamma} = 1$   
functional  
margin

Lagrangian

$$h(\vec{w}, b, \vec{\alpha}) = \underbrace{\frac{1}{2} \|\vec{w}\|^2}_{\text{"f"}} - \sum_{i=1}^n \alpha_i \underbrace{[y_i(\vec{w} \cdot \vec{x}_i + b) - 1]}_{\text{"g"}} \geq 0$$

$$\boxed{\nabla h = 0} \quad \text{primal vs dual}$$

$$p^* = \min_{\vec{w}, b} \left[ \max_{\vec{\alpha}, \alpha_i \geq 0} h(\vec{w}, b, \vec{\alpha}) \right]$$

"primal"

Sometimes

$$\boxed{p^* = d^*}$$

Switch max & min!

$$\geq \max_{\vec{\alpha}, \alpha_i \geq 0} \left[ \min_{\vec{w}, b} h(\vec{w}, b, \vec{\alpha}) \right] = d^*$$

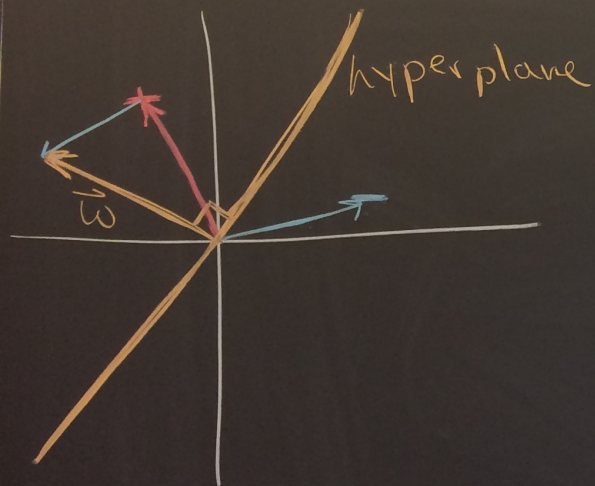
"dual"

# Solving the Dual

$$\nabla_{\vec{w}} h(\vec{w}, b, \vec{x}) = \vec{w} - \sum_{i=1}^n \alpha_i y_i \vec{x}_i = 0$$

$$\Rightarrow \boxed{\vec{w} = \sum_{i=1}^n \alpha_i y_i \vec{x}_i}$$

like  
perceptron  
updates!

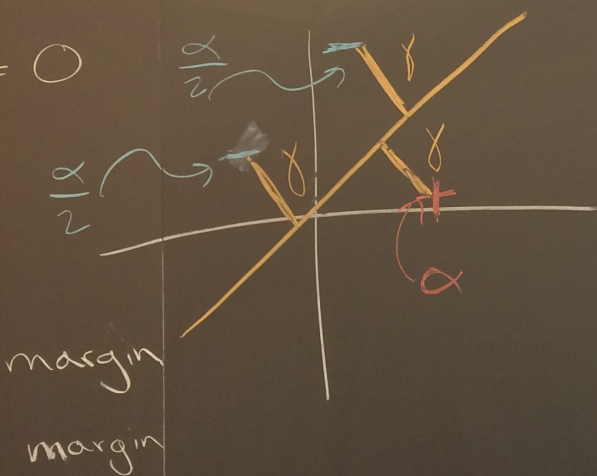


$$\frac{\partial h(\vec{w}, b, \vec{x})}{\partial b} = - \sum_{i=1}^n \alpha_i y_i = 0$$

$$\Rightarrow \boxed{\sum_{i=1}^n \alpha_i y_i = 0}$$

$\alpha_i = 0$  if  $\vec{x}_i$  not on margin  
 $\alpha_i > 0$  if  $\vec{x}_i$  is on margin

$$\boxed{\sum_{i: y_i=1} \alpha_i = \sum_{i: y_i=-1} \alpha_i}$$



$$h(\vec{w}, b, \vec{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \boxed{\vec{x}_i \cdot \vec{x}_j} = W(\vec{\alpha})$$

★

Notice: no more  $\vec{w}$ 's +  $b$ 's!

new dual optimization problem:

$$\begin{aligned} \max_{\vec{\alpha}} \quad & W(\vec{\alpha}) \quad \star \\ \text{s.t.} \quad & \alpha_i \geq 0 \\ \text{and} \quad & \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

dot product  
is an example  
of an inner  
product

intuition: measure  
of similarity

$$h(\vec{w}, b, \vec{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \boxed{\vec{x}_i \cdot \vec{x}_j} = W(\vec{\alpha})$$

★

Notice: no more  $\vec{w}$ 's +  $b$ 's!

new dual optimization problem:

$$\begin{array}{ll} \max_{\vec{\alpha}} & W(\vec{\alpha}) \\ \text{s.t.} & \alpha_i \geq 0 \\ \text{and} & \sum_{i=1}^n \alpha_i y_i = 0 \end{array} \quad \star$$

dot product  
is an example  
of an inner  
product

intuition: measure  
of similarity.

# Outline for March 27

- Finish Lagrangian for SVMs
- **Kernels**
- Soft-margin SVMs
- How to solve the optimization problem

# Kernel Idea

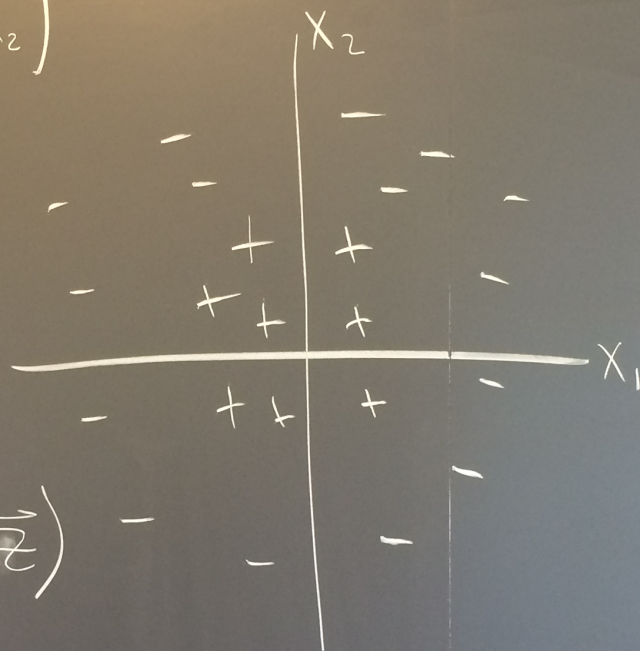
- By solving the dual form of the problem, we have seen how all computations can be done in terms of inner products between examples
- One example of an inner product is the dot product, which is the linear version of SVMs
- But there are many others!
- Intuition: if points are close together, their kernel function will have a large value (measure of similarity)

feature mapping

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\phi(\vec{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 + x_2^2 \end{bmatrix}$$

Kernel function

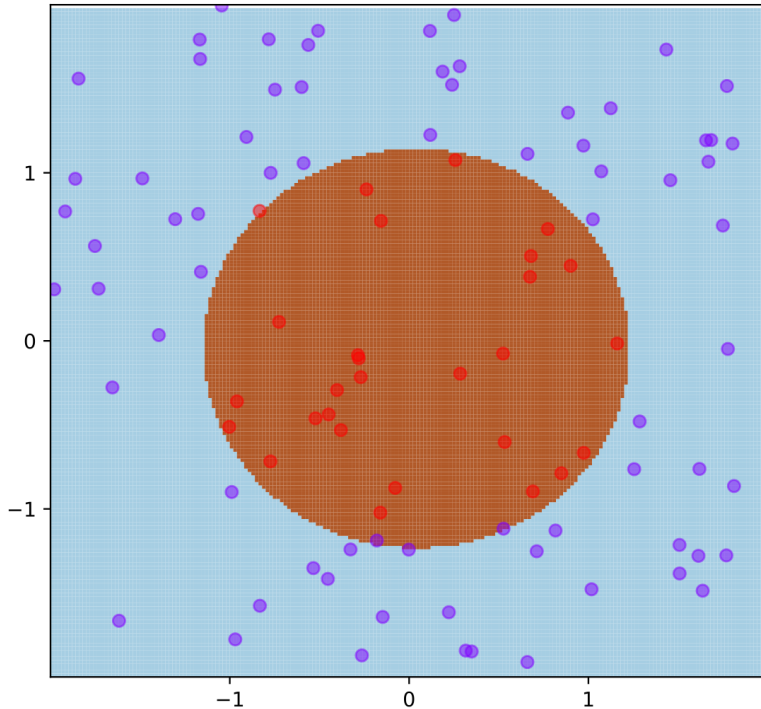


$$K(\vec{x}, \vec{z}) = \phi(\vec{x}) \cdot \phi(\vec{z})$$

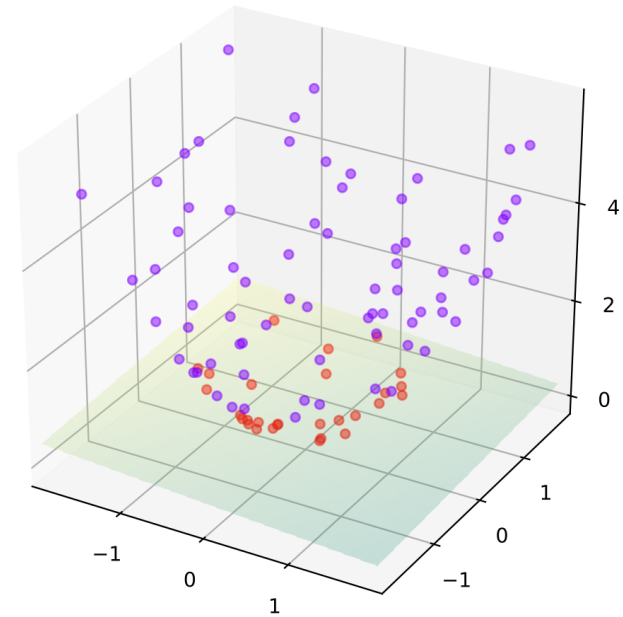
$$= \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 + x_2^2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \\ z_1^2 + z_2^2 \end{bmatrix} = \boxed{\vec{x} \cdot \vec{z} + \|\vec{x}\|^2 \|\vec{z}\|^2}$$

# Kernel Trick example

Feature mapping:  $\varphi(\mathbf{x}) = (x_1, x_2, x_1^2 + x_2^2)$



Original feature space



Mapping after applying kernel  
(can now find a hyperplane)

Kernel function:  $K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \bullet \mathbf{z} + \|\mathbf{x}\|^2 \|\mathbf{z}\|^2$

# Gaussian Kernel

- Gaussian kernel is near 0 when points are far apart and near 1 when they are similar
- Also called Radial Basis Function (RBF) kernel

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

# Gaussian Kernel

- Gaussian kernel is near 0 when points are far apart and near 1 when they are similar
- Also called Radial Basis Function (RBF) kernel

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

Often re-parametrized by  
gamma (different gamma!)

$$\gamma = \frac{1}{2\sigma^2}$$

$$K(\vec{x}, \vec{z}) = \exp\left(-\gamma\|\vec{x} - \vec{z}\|^2\right)$$

We will tune gamma as part of Lab 6

$$h(\vec{x}) = \text{sign}(\vec{w}^* \cdot \vec{x} + b)$$

$$\vec{w}^* = \sum_{i=1}^n \alpha_i^* y_i \vec{x}_i$$

$$= \text{sign}\left(\sum_{i=1}^n \alpha_i^* y_i \underbrace{\vec{x}_i \cdot \vec{x}}_{\text{Your Kernel}} + b\right)$$

Only need  
to consider  
the support vectors

function of choice

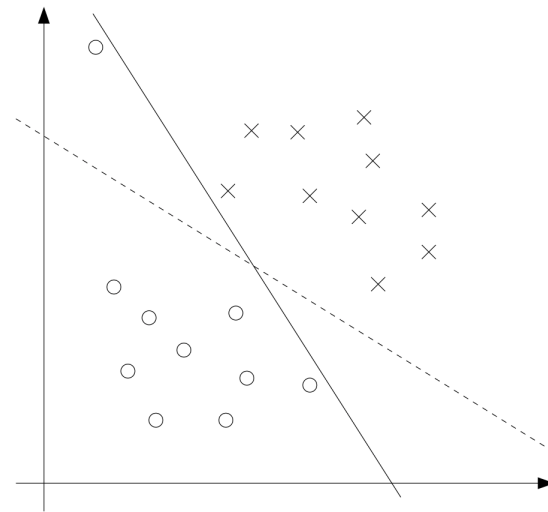
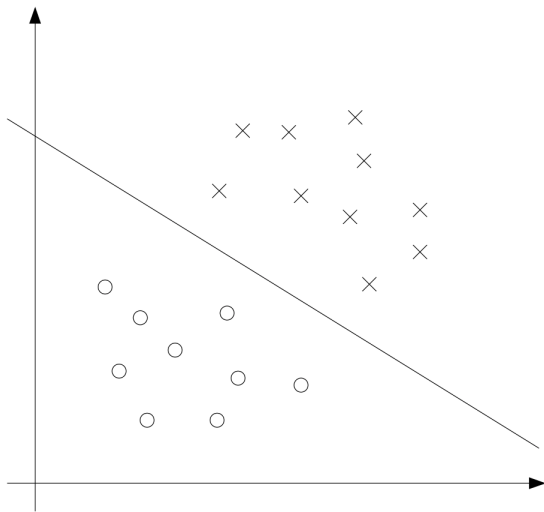
Kernel can be used for prediction as well!

# Outline for March 27

- Finish Lagrangian for SVMs
- Kernels
- **Soft-margin SVMs**

# Soft-margin SVMs (non-separable case)

- Idea: we will use regularization to add a cost for each point being incorrectly classified by the hyperplane
- Hopefully many costs will be 0, but we can accommodate a few outliers



# Soft-margin SVMs (non-separable case)

- New optimization problem with regularization
- We will tune the  $C$  parameter as part of Lab 6

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

$$\text{s.t.} \quad y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \dots, n$$

$$\text{and} \quad \xi_i \geq 0, \quad i = 1, \dots, n$$