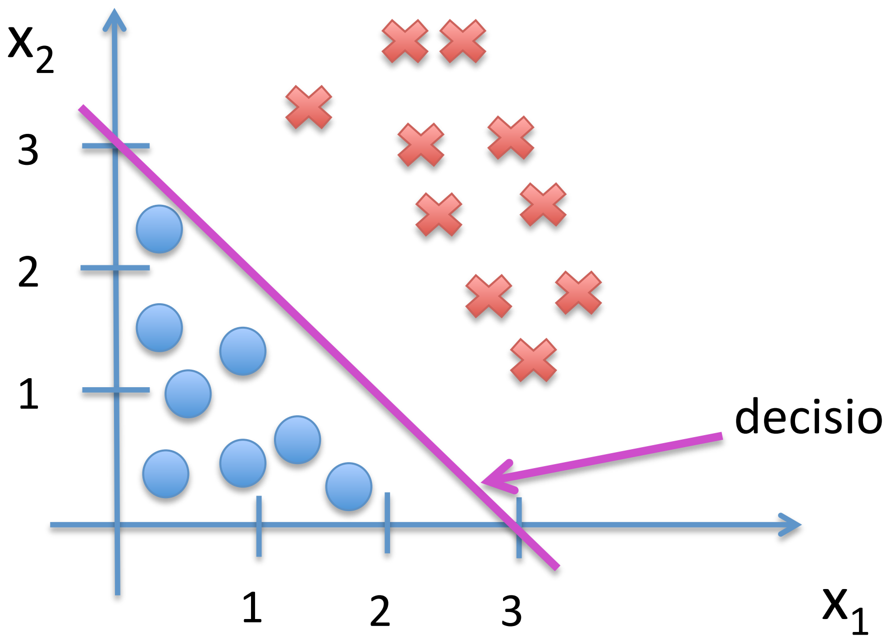
Machine Learning:
Support Vector Machines (SVMs), Part 2

Slides adapted from David Sontag, who adapted from Luke Zettlemoyer, Vibhav Gogate, and Carlos Guestrin

Kernel Trick

- Allows for non-linear decision boundaries
- Allows you to dramatically increase number of features without significantly increasing storage or computation
- Mathematical: Uses duality theory from convex optimization

Decision Boundary

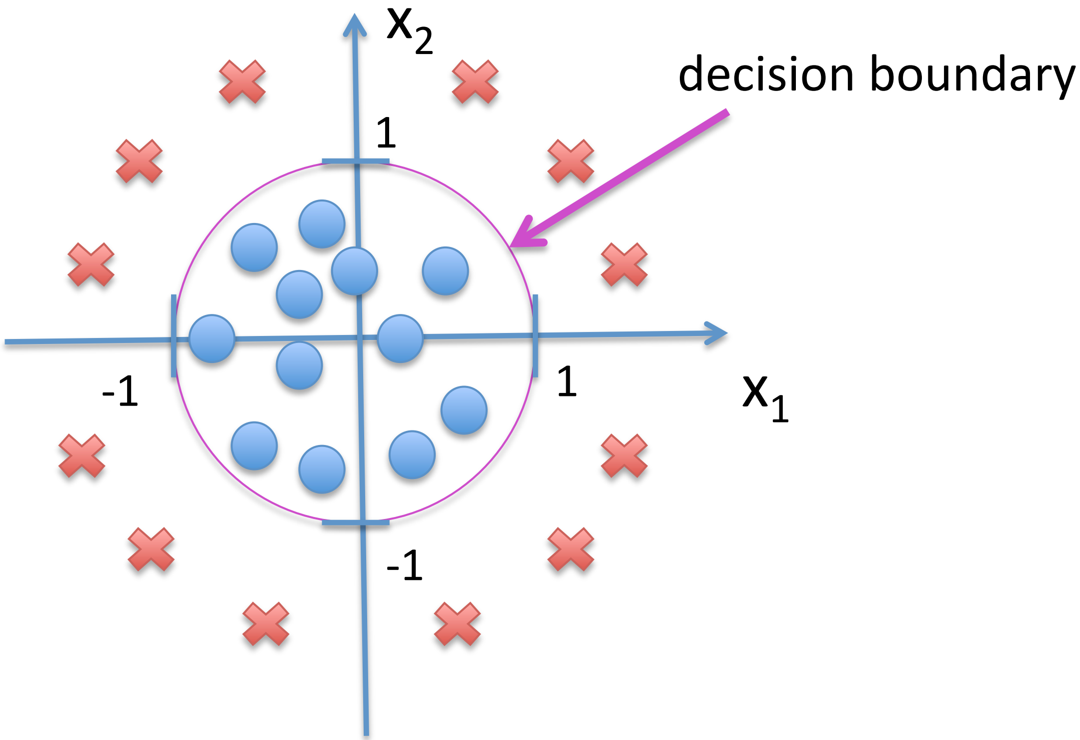


$$h_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2$$

$$\boldsymbol{w} = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

Predict "y=1" if $-3+x_1+x_2 \geq 0$

Non-linear decision boundaries



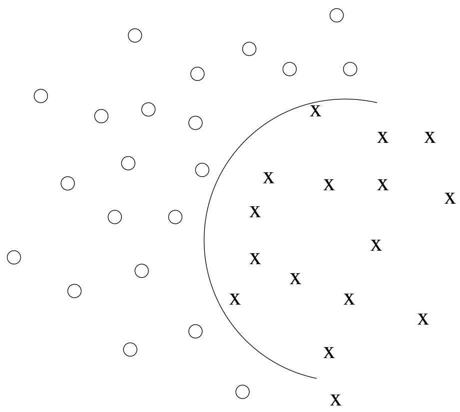
$$\boldsymbol{w} \in \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$h_w(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2$$

Predict "y=1" if $-1 + x_1^2 + x_2^2 \geq 0$

Non-linear decision region

$$b + w_1x_1 + w_2x_2 + \ldots + w_nx_n + w_{n+1}x_1x_2 + w_{n+2}x_1x_3 + \ldots + w_q e^{x1} > 0$$



Non-linear separator in the original $\mathbf{x}$-space

- Can potentially reduce errors
- But becomes much more difficult to solve non-linear optimization problem

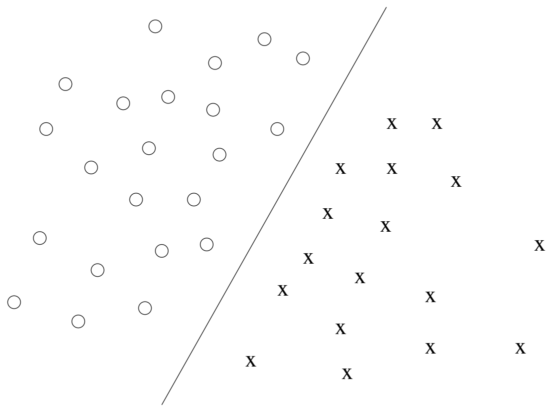
Consider using features that are functions of features

- For example, in addition to current features $x_1, x_2, \ldots, x_n$, use features like $x_1 x_2$, $x_1, x_3, \ldots, e^{x1}$, e^{x2}.
- New set of features:
  - $\varphi_1 = x_1$, $\varphi_2 = x_2$, $\varphi_n = x_n$, $\varphi_{n+1} = x_1 x_2$, $\varphi_{n+2} = x_1 x_3, \ldots.$
- New feature vector
  - $\boldsymbol{\varphi} = (x_1, x_2, \ldots, x_n, x_1 x_2, x_1 x_3, \ldots, e^{x1}, e^{x2}, \ldots)$

– From original data $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$, can create new data $\boldsymbol{\varphi}^{(1)}, \boldsymbol{\varphi}^{(2)}, \ldots, \boldsymbol{\varphi}^{(m)}$

Decision Region in ϕ space

$$b+w_1\varphi_1+\ldots+w_n\,\varphi_n+w_{n+1}\,\varphi_{n+1}+w_{n+2}\,\varphi_{n+2}+\ldots.\ w_Q\varphi_Q > 0$$



Linear separator in the feature ϕ-space

- This is now linear! Can use standard SVM methodology.

  $\text{Minimize}_{\mathbf{w},b,\xi}\ \mathbf{w}\bullet\mathbf{w} + C\,\Sigma_i\,\xi^{(i)}$
  subject to
  $(\mathbf{w}\bullet\boldsymbol{\varphi}(\mathbf{x}^{(i)})+b)y^{(i)} \geq 1 - \xi^{(i)}$ for all i

- But **w** and **φ** are now Q >> N dimensional.

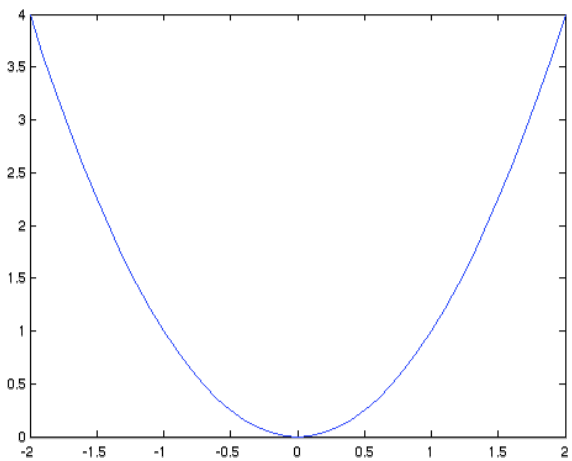
What’s Next!

- One of the most interesting and exciting advances in machine learning: Kernel trick
- Basic idea: for some classes of non-linear features, dimensionality doesn’t increase
- But first, a detour
  - Constrained optimization!

Constrained optimization

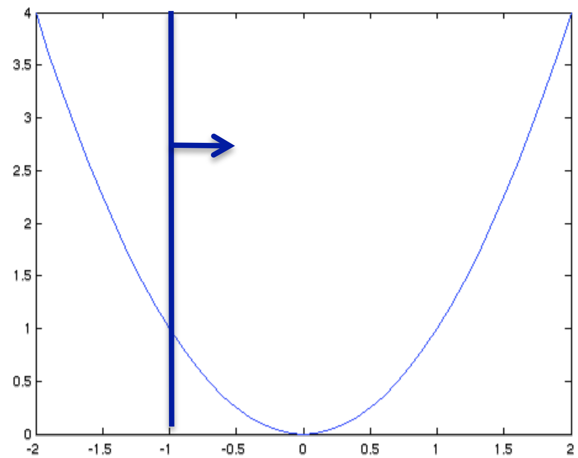
$$\min_x \ x^2$$
$$\text{s.t.} \quad x \geq b$$

No Constraint



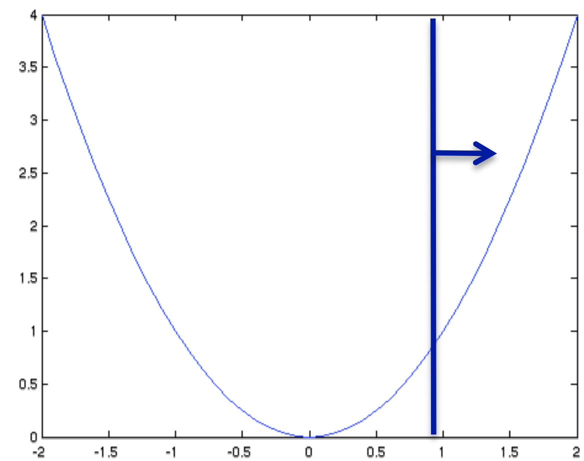
$x^*=0$

$x \geq -1$



$x^*=0$

$x \geq 1$

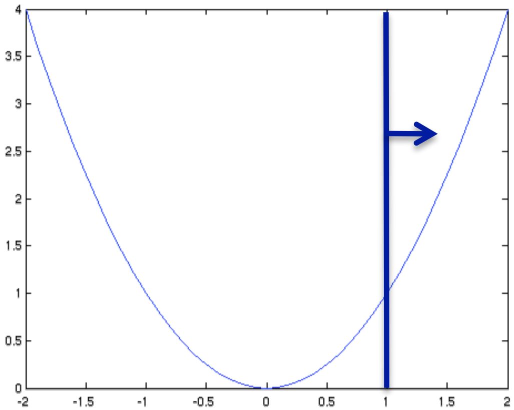


$x^*=1$

How do we solve with constraints?

→ Lagrange Multipliers!!!

Lagrange multipliers – Dual variables



$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$

Add Lagrange multiplier

Rewrite Constraint

Introduce Lagrangian (objective):

$$L(x, \alpha) = x^2 - \alpha(x - b)$$

We will solve:

$$\min_x \max_\alpha \; L(x, \alpha)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Add new constraint

Can easily show that if x*, α* optimal for min-max problem, then x* is optimal for original problem.

Dual SVM derivation (hard margin SVM)

Original optimization problem:

$$\text{Minimize } \tfrac{1}{2}\, \mathbf{w} \bullet \mathbf{w}$$
$$\text{s.t. } (\mathbf{w} \bullet \mathbf{x}^{(i)} + b)\, y^{(i)} \geq 1 \text{ for all } i$$

Rewrite constraints

One Lagrange multiplier per example

Lagrangian:

$$L(\mathbf{w}, \boldsymbol{\alpha}) = \tfrac{1}{2}\, \mathbf{w} \bullet \mathbf{w} - \Sigma_i\, \alpha_i \left[(\mathbf{w} \bullet \mathbf{x}^{(i)} + b)\, y^{(i)} - 1 \right]$$
$$\alpha_i \geq 0 \text{ for all } i.$$

Our goal now is to solve:

$$\min_{\mathbf{w},b} \max_{\boldsymbol{\alpha} \geq 0} L(\mathbf{w}, \alpha)$$
$$= \max_{\boldsymbol{\alpha} \geq 0} \min_{\mathbf{w},b} L(\mathbf{w}, \alpha)$$

Equivalent dual problem

$$\max_{\boldsymbol{\alpha}\geq 0} \min_{\mathbf{w},b} \;\; \tfrac{1}{2}\, \mathbf{w}\bullet\mathbf{w} - \Sigma_i\, \alpha_i \left[(\mathbf{w}\bullet\mathbf{x}^{(i)}+b)\, y^{(i)}-1\right]$$

- Take the partial derivative of $L(\mathbf{w},\boldsymbol{\alpha})$ w.r.t $w_1,..w_n,b$ and set to zero:

$$\mathbf{w} = \Sigma_i\, \alpha_i\, y^{(i)}\mathbf{x}^{(i)}$$

$$\Sigma_i\, \alpha_i\, y^{(i)} = 0$$

- Substitute back in and simplify

(dual) $\max_{\boldsymbol{\alpha}\geq 0} \Sigma_i\, \alpha_i - \tfrac{1}{2}\, \Sigma_{i,j} y^{(i)}y^{(j)}\, \alpha_i\, \alpha_j\, \mathbf{x}^{(i)}\bullet\, \mathbf{x}^{(j)}$

s.t. $\Sigma_i\, \alpha_i\, y^{(i)} = 0$

From dual to primal

(dual) $\max_{\boldsymbol{\alpha} \geq 0} \Sigma_i \alpha_i - \frac{1}{2} \Sigma_{i,j} y^{(i)} y^{(j)} \alpha_i \alpha_j \mathbf{x}^{(i)} \bullet \mathbf{x}^{(j)}$
s.t. $\Sigma_i \alpha_i y^{(i)} = 0$

- After solving dual, can get optimal primal solution:

$\mathbf{w} = \Sigma_i \alpha_i y^{(i)} \mathbf{x}^{(i)}$

b = see Andrew Ng notes

Predict with y ← sign [$\mathbf{w} \bullet \mathbf{x}$ + b]

$= \text{sign} [\Sigma_i \alpha_j y^{(i)} (\mathbf{x} \bullet \mathbf{x}^{(i)}) + b]$

Features of features

$$\phi(x) = \begin{pmatrix} x^{(1)} \\ \dots \\ x^{(n)} \\ x^{(1)}x^{(2)} \\ x^{(1)}x^{(3)} \\ \dots \\ e^{x^{(1)}} \\ \dots \end{pmatrix}$$

Feature vector: $\boldsymbol{\phi}(\mathbf{x}^{(i)})$

Weight vector: **w**

Feature vector and weight vector super-high dimensional

Minimize $||\mathbf{w}||^2$

subject to:

$y_t(\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}^{(i)}) + b) \geq 1$ for all i

Consider Dual

$$\phi(x) = \begin{pmatrix} x^{(1)} \\ \ldots \\ x^{(n)} \\ x^{(1)}x^{(2)} \\ x^{(1)}x^{(3)} \\ \ldots \\ e^{x^{(1)}} \\ \ldots \end{pmatrix}$$

Maximize over $\boldsymbol{\alpha} \geq 0$:

$$\Sigma_i \alpha_i - \frac{1}{2} \Sigma_{i,j} y^{(i)} y^{(j)} \alpha_i \alpha_j K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$$

$$K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) := \boldsymbol{\phi}(\mathbf{x}^{(i)}) \bullet \boldsymbol{\phi}(\mathbf{x}^{(j)})$$

$$\text{s.t. } \Sigma_i \alpha_i y^{(i)} = 0$$

$$\mathbf{w} = \Sigma_i \alpha_i y^{(i)} \boldsymbol{\phi}(\mathbf{x}^{(i)})$$

Still looks hard, since we still have to deal with huge-dimensional $\boldsymbol{\phi}(\mathbf{x}^{(i)})$

Kernel trick: for some $\boldsymbol{\Phi}(\mathbf{x})$, can evaluate kernel $K(x,z)$ directly without evaluating $\boldsymbol{\Phi}(\mathbf{x})$!

Example

Consider following feature of feature vector:

$$\phi(x) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \end{bmatrix}.$$

In this example, we increased the number of features from O(n) to O(n^2); but the dual is:

$$\max_{\boldsymbol{\alpha}} \Sigma_i \alpha_i - \tfrac{1}{2} \Sigma_{i,j} y^{(i)} y^{(j)} \alpha_i \alpha_j (\mathbf{x}^{(i)} \bullet \mathbf{x}^{(j)})^2$$

$$\text{s.t. } \Sigma_i \alpha_i y^{(i)} = 0$$

$$y \leftarrow \text{sign} [\Sigma_i \alpha_i y^{(i)} (\mathbf{x} \bullet \mathbf{x}^{(i)})^2 + b]$$

$$\phi(\mathbf{x})\ \phi(\mathbf{z}) = (\mathbf{x} \bullet \mathbf{z})^2$$
$$:= K(\mathbf{x}, \mathbf{z})$$

Only have to deal with feature vectors of length n ! Thus, obtain order of magnitude of new features & nonlinear decision boundary for free!

Story summary

- Suppose you want to use lots of non-linear features $\boldsymbol{\phi}(\mathbf{x})$. Computationally impossible.
- But suppose $K(\mathbf{x},\mathbf{z}) = \boldsymbol{\phi}(\mathbf{x})\bullet\boldsymbol{\phi}(\mathbf{z})$ can be evaluated without explicitly evaluating $\boldsymbol{\phi}(\mathbf{x})$.
- Then solving the dual for the α_i's and then doing classification is tractable.
- Research problem: find "kernels" $K(\mathbf{x},\mathbf{z})$ that are easy to evaluate and can be expressed as $K(\mathbf{x},\mathbf{z}) = \boldsymbol{\phi}(\mathbf{x})\bullet\boldsymbol{\phi}(\mathbf{z})$

Quadratic kernel

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= (\mathbf{x}^T\mathbf{z} + c)^2 = \left(\sum_{j=1}^{n} x^{(j)} z^{(j)} + c\right)\left(\sum_{\ell=1}^{n} x^{(\ell)} z^{(\ell)} + c\right) \\
&= \sum_{j=1}^{n}\sum_{\ell=1}^{n} x^{(j)} x^{(\ell)} z^{(j)} z^{(\ell)} + 2c\sum_{j=1}^{n} x^{(j)} z^{(j)} + c^2 \\
&= \sum_{j,\ell=1}^{n} (x^{(j)} x^{(\ell)})(z^{(j)} z^{(\ell)}) + \sum_{j=1}^{n} (\sqrt{2c}x^{(j)})(\sqrt{2c}z^{(j)}) + c^2,
\end{aligned}
$$

Feature mapping given by:

$$
\mathbf{\Phi}(\mathbf{x}) = [x^{(1)2}, x^{(1)}x^{(2)}, ..., x^{(3)2}, \sqrt{2c}x^{(1)}, \sqrt{2c}x^{(2)}, \sqrt{2c}x^{(3)}, c]
$$

[Cynthia Rudin]

Common kernels

- Polynomials of degree exactly *d*

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z})^d$$

- Polynomials of degree up to *d*

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + 1)^d$$

- Gaussian kernels

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\frac{\|\mathbf{x} - \mathbf{z}\|_2^2}{2\sigma^2})$$

- And many others!

Soft SVM with kernels

Maximize:

$$\Sigma_i \alpha_i - \tfrac{1}{2} \Sigma_{i,j} y^{(i)} y^{(j)} \alpha_i \alpha_j K(\mathbf{x}^{(i)},\mathbf{x}^{(j)})$$

s.t. $\Sigma_i \alpha_i y^{(i)} = 0$

$0 \le \alpha_i \le C \quad \text{for } i$

$$K(\mathbf{x},\mathbf{z}) := \boldsymbol{\phi}(\mathbf{x}) \bullet \boldsymbol{\phi}(\mathbf{z})$$

- Almost the same as hard-margin dual
- Once again, tractable for kernels that are easy to evaluate. Never compute features explicitly!!!
- $O(m^2)$ computation in size of dataset m to compute objective
  - much work on speeding up