Support Vector Machines (SVMs)

References:

Cristianini & Shawe-Taylor book; Vapnik's book; and "A Tutorial on Support Vector Machines for Pattern Recognition" by Burges, in *Datamining and Knowledge Discovery 2, 1998 (check citeseer)*

kernel-machines.org

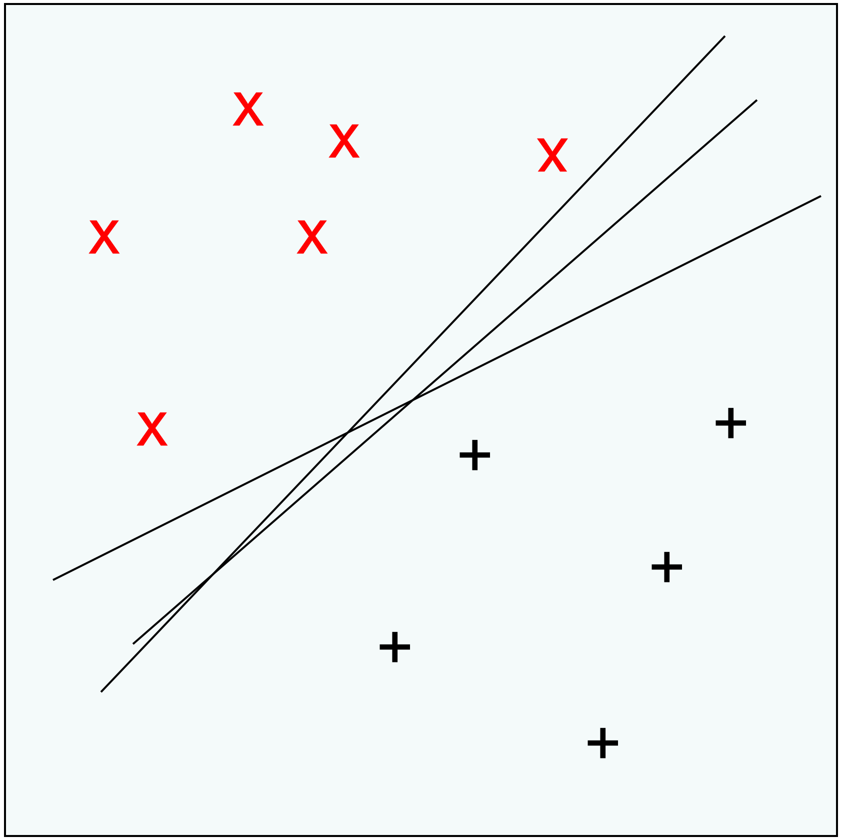
SVMs

- Combines learning and optimization theory
- *Exactly* solve optimization (as opposed to ANN or Decision Trees)
- Allows "kernel trick" to get more features almost for free
- Soft margin relaxes linear separable assumption

Basic Idea: (separable case)

- Which linear threshold hypothesis to use?
- Pick one with biggest ***margin*** (gap)
  - Intuitively good, safe against small errors
  - Some supporting theory
  - Empirically works well
- Leads to a quadratic programming problem

(Vapnik 95, Cortes and Vapnik 95)

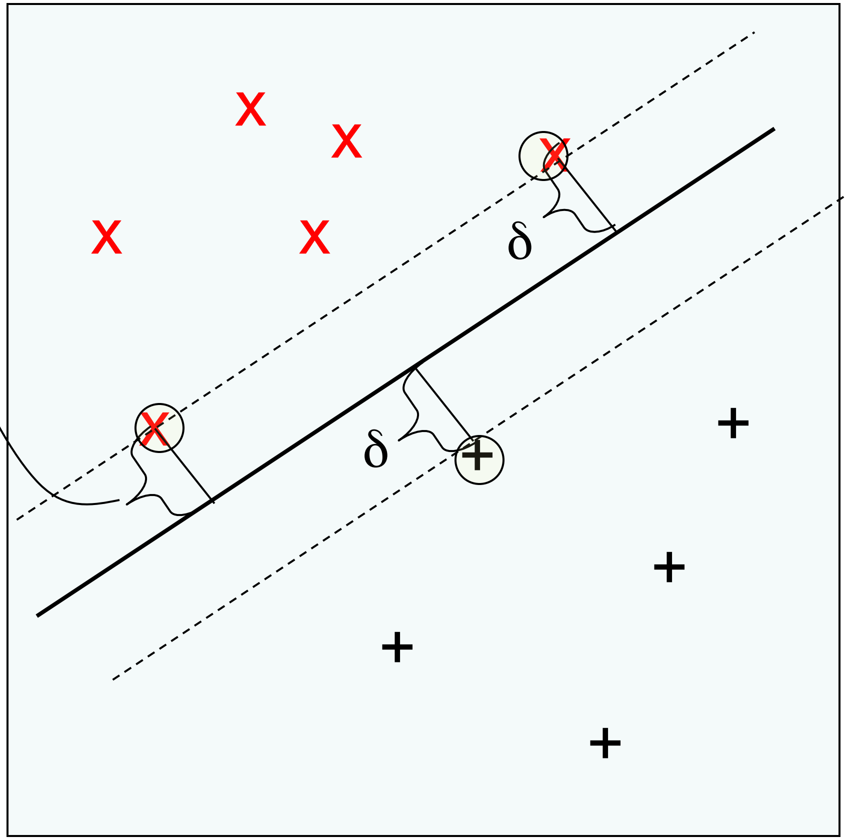


Basic Idea:

- Pick one with biggest ***margin*** (gap) δ
- Leads to a quadratic programming problem

(Vapnik 95, Cortes and Vapnik 95)

- ***Support Vectors*** are the points with smallest margin (closest to boundary, circled)
- hypothesis stability: it only changes if support vectors change



Good generalization

- If $f(\boldsymbol{x}) = \sum w_j x_j + b$ and $\sum w_j^2 = 1$ and large margin, then it (expects) to generalize well

  generalization error: $\tilde{O}\left(\frac{R^2}{N\delta^2} + \frac{\log(1/\text{conf})}{N}\right)$

R = radius of support, N = # examples, delta = margin

this is independent of dimension

(# of features) !

SVM applications

- Text classification - (Joachims, used outlier removal when a_i too large, Joachims also author of SVM-light)
- Object detection (e.g. Osuna et.al., for faces)
- Hand written digits - few support vectors
- Bioinformatics - protein homology, micro-array
- May be best current learning method

See Cristianini&Shaw-Taylor book, and [www. kernel-machines.org](http://www.kernel-machines.org) for more info and references

SVM Mathematically

- Find $\boldsymbol{w}$, b, δ such that:

$$\boldsymbol{w} \cdot \boldsymbol{x_i} + b \geq \delta \quad \text{when } y_i = +1$$

$$\boldsymbol{w} \cdot \boldsymbol{x_i} + b \leq -\delta \quad \text{when } y_\mathrm{i} = -1$$

  and δ as big as possible
- Scaling issue: fix by setting δ=1 and finding shortest $\boldsymbol{w}$:

$$\min_{\boldsymbol{w},b} ||\boldsymbol{w}||^2 \text{ subject to}$$

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1 \quad \text{for all examples } (\boldsymbol{x_i}, y_i)$$

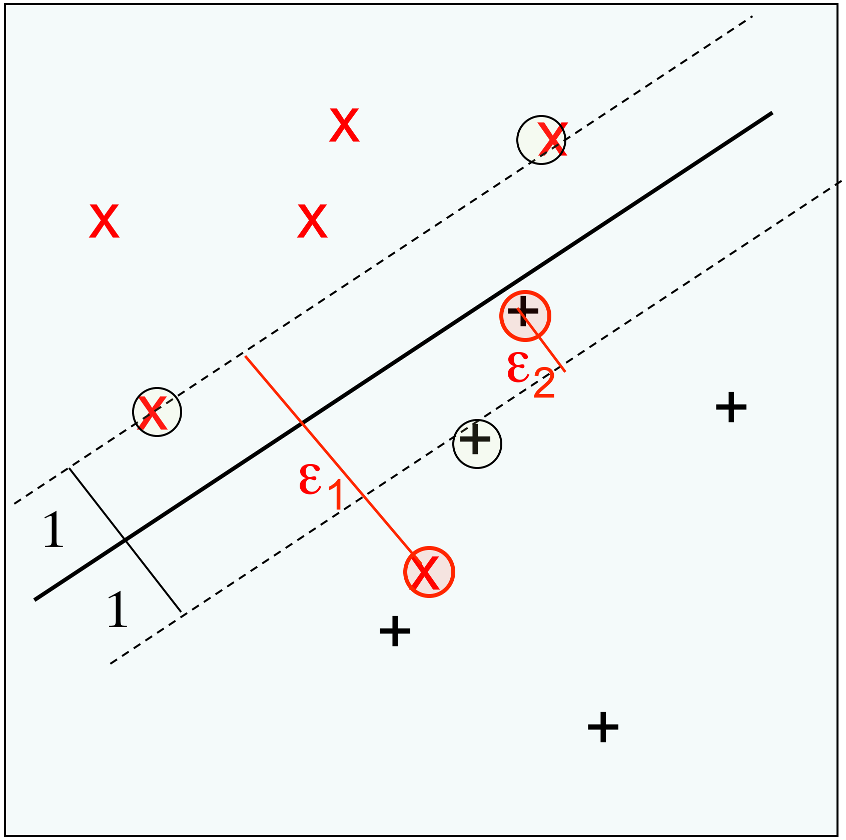
- This is a quadratic programming problem, packages exist (also SMO algorithm)

More Math $^{\text{not so}}$ (Quickly)

- LaGrange multipliers, dual problem, …, magic, …
- The ***w*** vector is a linear combination of support vectors: $\boldsymbol{w} = \sum_{\text{sup vects i}} a_i \boldsymbol{x_i}$
- Predict on ***x*** based on $\boldsymbol{w} \cdot \boldsymbol{x} \geq b$, or equivalently $\sum_{\text{sup vects i}} a_i (\boldsymbol{x_i} \cdot \boldsymbol{x}) \geq b$
- Key idea 1: predictions (and finding a_i' s) depend only on dot products (also true for least squares and perceptron)

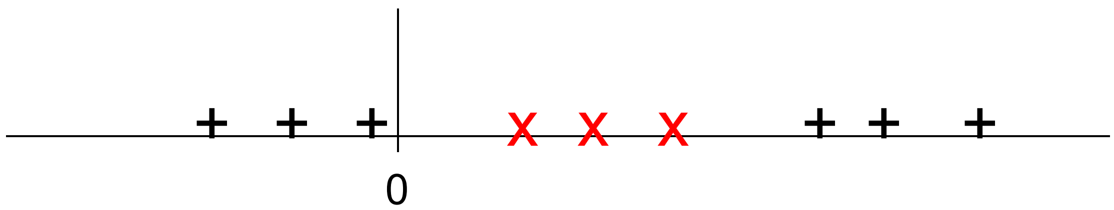
Soft Margin for noise

- Allow margin errors
- $\varepsilon_i \geq 0$ measures amount of error on $\boldsymbol{x_i}$
- Want to minimize both $\boldsymbol{w \cdot w}$ and $\sum_i \varepsilon_i$
- Need tradeoff parameter
- Still Quadratic programming, math messier

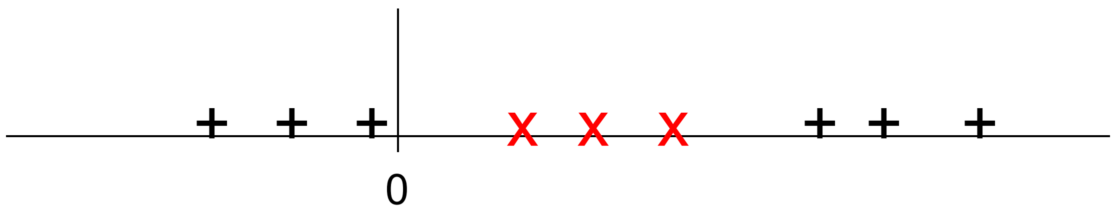


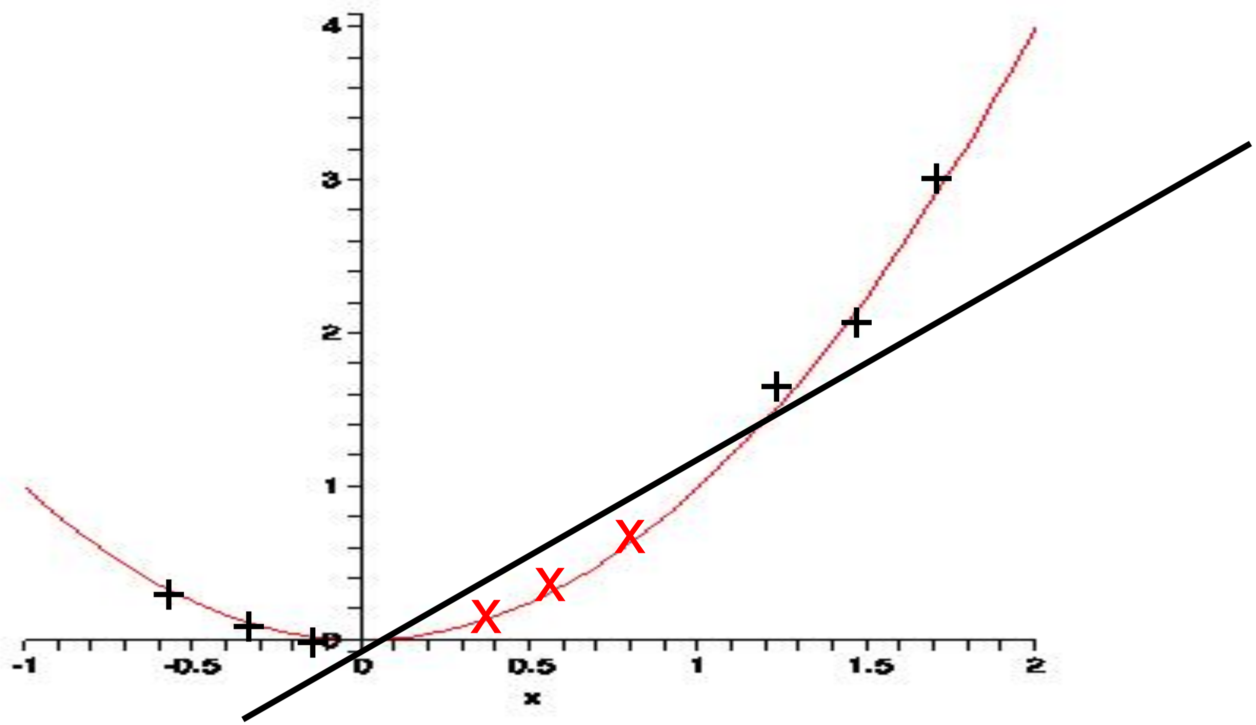
Kernel functions

- Don't need to use dot product, can use any "dot-product like" function K($\boldsymbol{x}$,$\boldsymbol{x'}$)
- Example: K($\boldsymbol{x}$,$\boldsymbol{x'}$) = $(\boldsymbol{x} \cdot \boldsymbol{x'}+1)^2$
- In 1-dimension: K(x,x') = $x^2x'^2 + 2xx' + 1$ = $(x^2, \sqrt{2}\,x, 1) \cdot (x'^2, \sqrt{2}\,x', 1)$, get quadratic term too (extra feature)
- How can this help?



Want to classify these points with a SVM,
No hyper-plane (threshold) has good margin


0


4
3
2
1
0
-1
-0.5
0
0.5
1
1.5
2
x

Kernel trick:

- Kernel functions embed the low dimensional original space into a higher dimensional one, like the film of a soap bubble
- Although sample is not linearly separable in the original space, the higher-dimensional embedding might be
- Get embedding and “extra” features for free (almost)
- General trick for any dot-product based algorithm (e.g. Perceptron)

Common Kernels

- Polynomials: $K(\boldsymbol{x},\boldsymbol{x'}) = (\boldsymbol{x} \cdot \boldsymbol{x'} + 1)^d$
  - Gets new features like $x_1 x_3$
- Radial Basis function (Gaussian):

  $K(\boldsymbol{x},\boldsymbol{x'}) = \exp(-||\boldsymbol{x} - \boldsymbol{x'}||^2 / 2\sigma^2)$
- Sigmoid: $K(\boldsymbol{x},\boldsymbol{x'}) = \tanh(a\,(\boldsymbol{x} \cdot \boldsymbol{x'}) - b)$
- Also special purpose string and graph kernels

Which to use and parameters? cross validation can help, <u>RBF Kernels powerful</u>

What can be a Kernel function?

- K($\boldsymbol{x}$,$\boldsymbol{z}$) is a (Mercer) kernel function if K($\boldsymbol{x}$,$\boldsymbol{z}$) = $\phi(\boldsymbol{x}) \bullet \phi(\boldsymbol{z})$

  implies symmetric

  for some feature transformation $\phi()$
- K($\boldsymbol{x}$,$\boldsymbol{z}$) is a Mercer kernel function if the matrix $\mathbf{K} = (k(\boldsymbol{x}_i,\boldsymbol{x}_j))_{i,j}$ is positive semi-definite for all subsets $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}$ of instance space
- Vapnik showed expected test error bound: (expected # support vectors) / (# examples)

Gaussian (RBF) Kernels

- Act like weighted nearest neighbor on support vectors
- Allow very flexible hypothesis space
- Mathematically a dot product in an "infinite dimensional" space
- SVMs unify LTU (linear kernel) and nearest neighbor (Gaussian kernel)

recall $\boldsymbol{w} = \Sigma\, \alpha_i \boldsymbol{x}_i$ what is the prediction?

Multiclass Kernel Machines

- 1-vs-all
- Pairwise separation (all pairs)
- Error-Correcting Output Codes
- Single multiclass optimization

$$\min \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{w}_i\|^2 + C\sum_i\sum_t \xi_i^t$$

subject to

$$\mathbf{w}_{z^t}{}^T\mathbf{x}^t + w_{z^t 0} \geq \mathbf{w}_i{}^T\mathbf{x}^t + w_{i0} + 2 - \xi_i^t,\ \forall i \neq z^t,\ \xi_i^t \geq 0$$

t superscript = example #

z^t = label of example t

i = class #

SVM for Regression

Use a linear model (possibly kernelized)

$$f(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+w_0$$

Use the <u>ε-(in)sensitive</u> error function

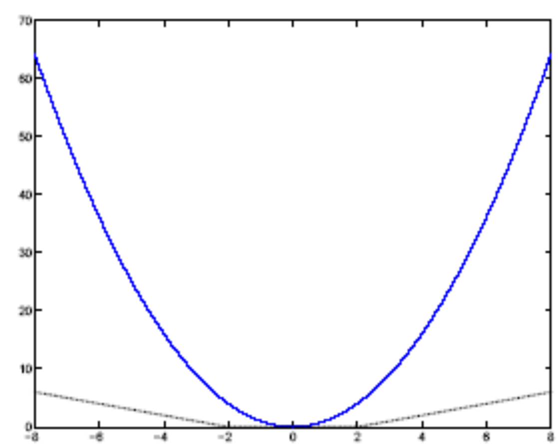
$$E_\varepsilon\left(y,f\left(\mathbf{x}\right)\right)=\begin{cases}0 & \text{if } \left|y-f\left(\mathbf{x}\right)\right|<\varepsilon \\ \left|y-f\left(\mathbf{x}\right)\right|-\varepsilon & \text{otherwise}\end{cases}$$

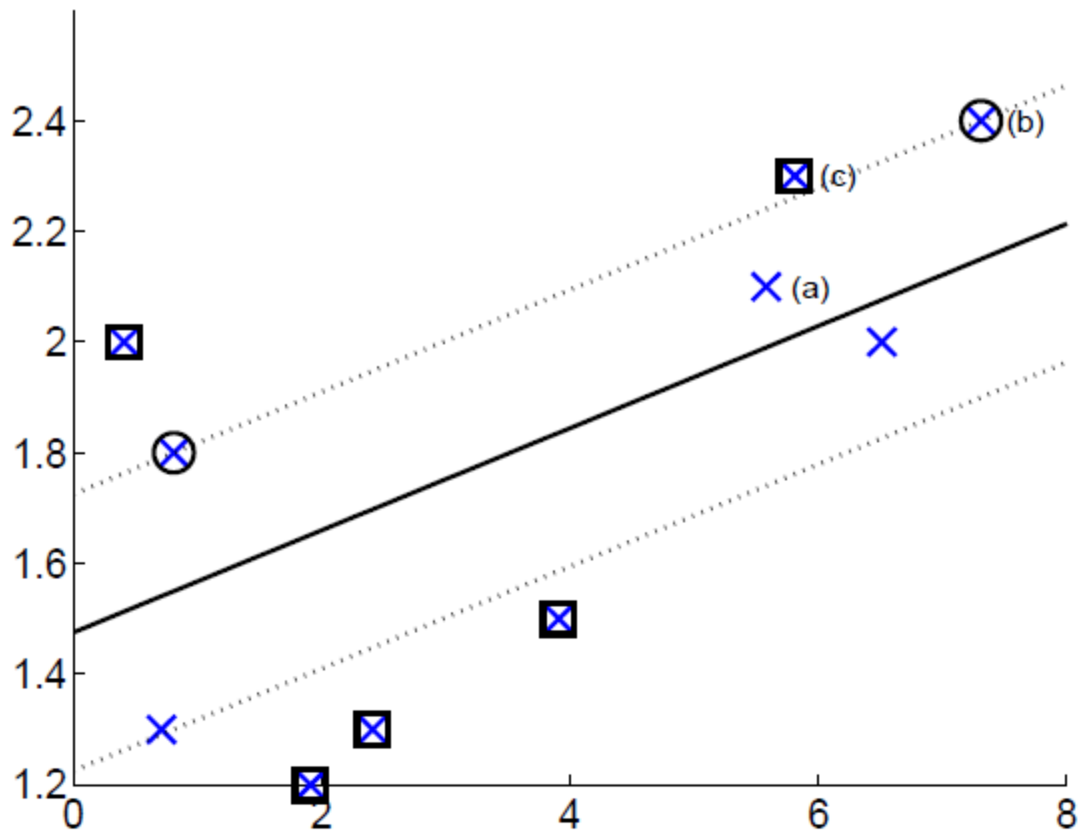
$$\min\frac{1}{2}\|\mathbf{w}\|^2+C\sum_t\left(\xi_+^t+\xi_-^t\right)$$

$$y_n-\left(\mathbf{w}^T\mathbf{x}_n+w_0\right)\le\varepsilon+\xi_n^+$$

$$\left(\mathbf{w}^T\mathbf{x}_n+w_0\right)-y_n\le\varepsilon+\xi_n^-$$

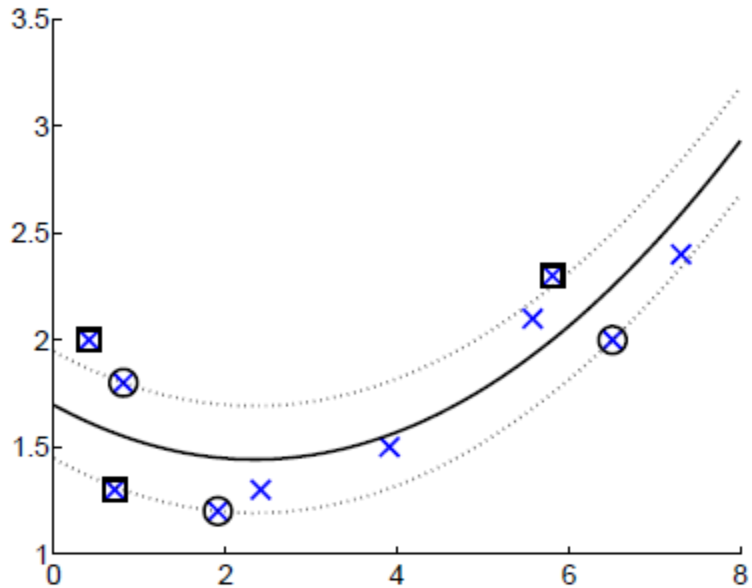
$$\xi_n^-,\xi_n^+\ge 0$$



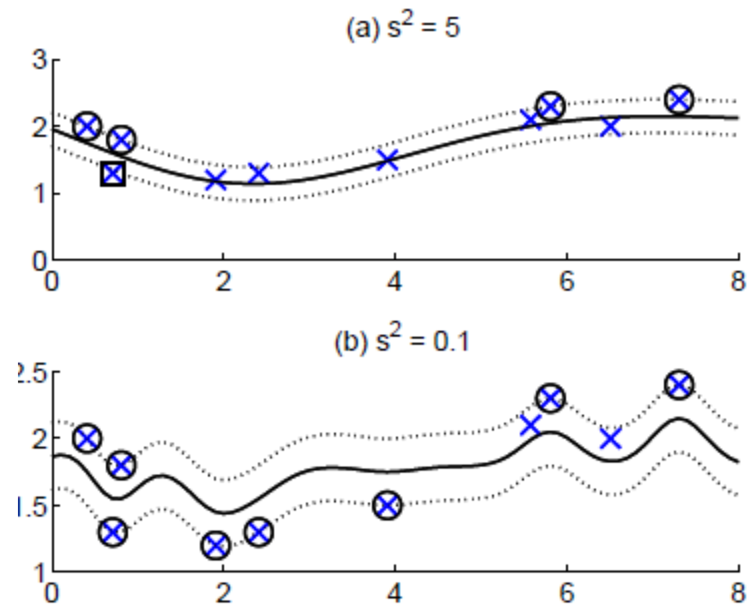

2.4
2.2
2
1.8
1.6
1.4
1.2
0
2
4
6
8
(a)
(b)
(c)

Kernel Regression

- Polynomial kernel



- Gaussian kernel



One-Class Kernel Machines

- Consider a sphere with center ***a*** and radius *R*

$$\min R^2 + C\sum_t \xi^t$$

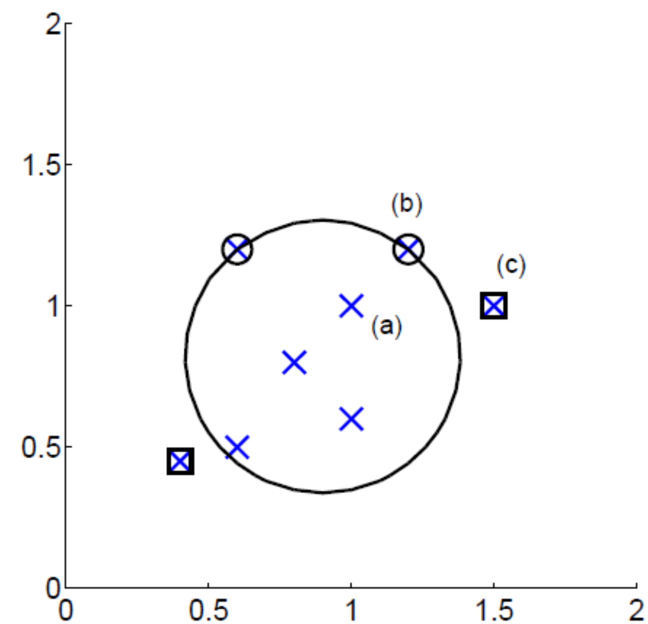
subject to

$$\left\|\mathbf{x}^t - a\right\| \le R^2 + \xi^t, \qquad \xi^t \ge 0$$

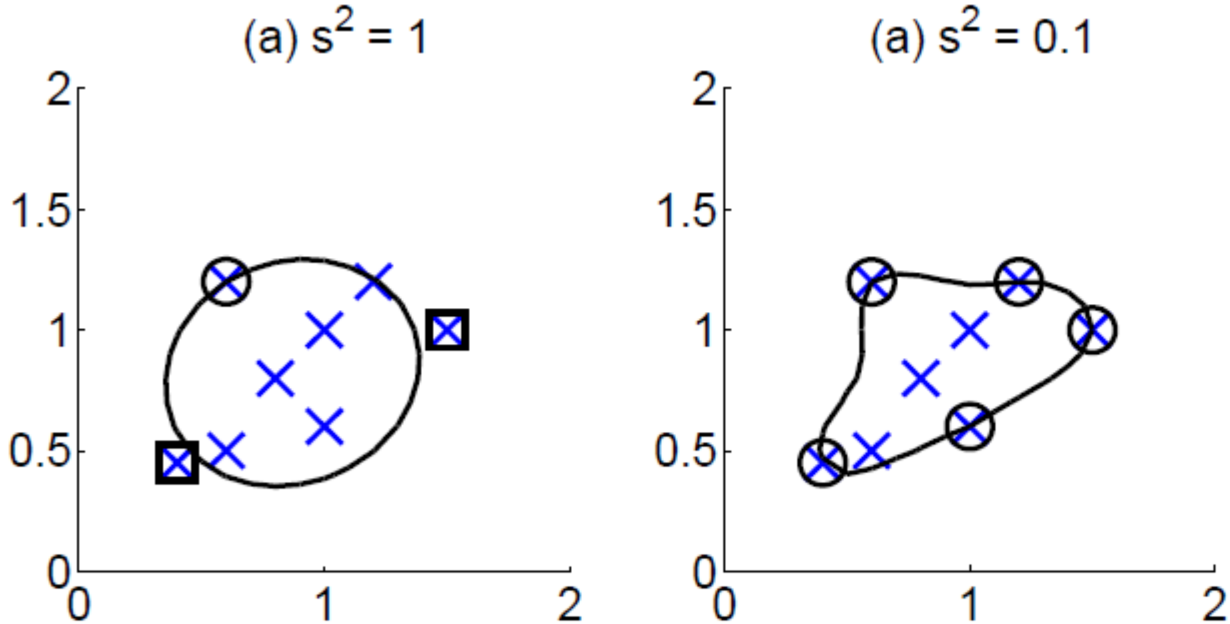
$$L_d = \sum_t \alpha^t \boxed{\left(x^t\right)^T x^t} - \sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \boxed{\left(x^t\right)^T x^s}$$

subject to

$$0 \le \alpha^t \le C, \qquad \sum_t \alpha^t = 1$$



One-class SVM with Gaussian kernels (s is sigma)



SVM Summary

- Model: very flexible, but must pick many parameters (kernel, kernel parameters, trade-off)
- Data: Numeric (depending on kernel)
- Interpretable? Yes for dot product, pretty pictures for Gaussian kernels in low dimensions
- Missing values? No
- Noise/outliers? Very good (softmargin)
- Irrelevant features? Yes for dot product with abs value penalty
- Comp. efficiency? quadratic in # examples, but "exact" optimization rather than approximate (like ANN, decision trees) Chunking techniques and other optimizations (SMO iteratively optimize a_i's over pairs, SGD, PEGASOS, etc.)