SVMs: Duality and Kernel Trick

Machine Learning - 10601

Geoff Gordon, MiroslavDudík

[partly based on slides of Ziv-Bar Joseph]

http://www.cs.cmu.edu/~ggordon/10601/

November 18, 2009

SVMs as quadratic programs

Two optimization problems: For the separable and non separable cases

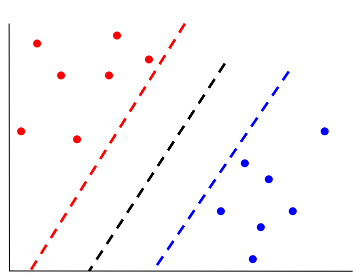
$$\min_{\mathbf{w},b} \frac{\mathbf{w}^T\mathbf{w}}{2}$$

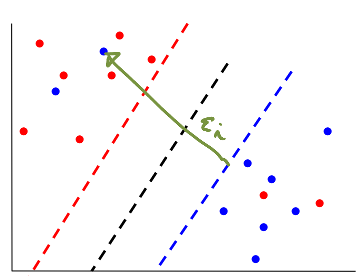
$(w^Tx_i+b)y_i \geq 1$

$$\min_{\mathbf{w},b,\varepsilon_i} \frac{\mathbf{w}^T\mathbf{w}}{2} + \sum_{i=1}^{n} C\varepsilon_i$$

$(w^Tx_i+b)y_i \geq 1- \varepsilon_i$

$\varepsilon_i \geq 0$





Dual for separable case

$$\min_{\mathbf{w},b} \frac{\mathbf{w}^{\mathsf{T}}\mathbf{w}}{2}$$

$(w^T x_i + b)y_i - 1 \geq 0$ (α_i)

#constraints
= #data pts
= #dual vars

Dual for separable case

$$\min_{\mathbf{w},b} \frac{\mathbf{w}^{\mathsf{T}}\mathbf{w}}{2}$$

$(w^T x_i + b)y_i - 1 \geq 0$ (α_i)

Lagrangean

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{\mathbf{w}^{\mathsf{T}}\mathbf{w}}{2} - \sum_i \alpha_i \left((\mathbf{w}^{\mathsf{T}}\mathbf{x}_i + b) y_i - 1 \right)$$

Dual for separable case

$$\min_{\mathbf{w},b} \frac{\mathbf{w}^\mathsf{T}\mathbf{w}}{2}$$

$(w^T x_i + b) y_i - 1 \geq 0$ (α_i)

EQUIVALENT

Lagrangean

$$L(\mathbf{w},b,\boldsymbol{\alpha}) = \frac{\mathbf{w}^\mathsf{T}\mathbf{w}}{2} - \sum_i \alpha_i \left((\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) y_i - 1 \right)$$

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

convex in w, b
concave in α

Dual for separable case

$$\min_{\mathbf{w},b} \frac{\mathbf{w}^\mathsf{T}\mathbf{w}}{2}$$

$(w^T x_i + b) y_i - 1 \geq 0$ (α_i)

Lagrangean

$$L(\mathbf{w},b,\boldsymbol{\alpha}) = \frac{\mathbf{w}^\mathsf{T}\mathbf{w}}{2} - \sum_i \alpha_i \left((\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) y_i - 1 \right)$$

KARUSH
-KUHN-TUCKER
KKT
conditions

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

optimality of α_i

for all i:

$$\alpha_i \geq 0$$

$$\underbrace{\alpha_i}_{\text{MULT.}} \underbrace{\left((\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) y_i - 1 \right)}_{\text{CONSTR}} = 0$$

optimality of w and b

$$\left.\begin{aligned} \mathbf{w} &= \sum_i \alpha_i \mathbf{x}_i y_i \\ \sum_i \alpha_i y_i &= 0 \end{aligned}\right\} \frac{\partial L}{\partial w} = 0, \quad \frac{\partial L}{\partial b} = 0$$

Dual for separable case

Dual formulation

$$\max_{\boldsymbol{\alpha}\geq 0}\min_{\mathbf{w},b}\left(\frac{\mathbf{w}^{\mathsf{T}}\mathbf{w}}{2}-\sum_i \alpha_i\left((\mathbf{w}^{\mathsf{T}}\mathbf{x}_i+b)y_i-1\right)\right)$$

Optimality conditions
(KKT conditions)

$$\mathbf{w}=\sum_i \alpha_i \mathbf{x}_i y_i$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0$$

$$\alpha_i\left((\mathbf{w}^{\mathsf{T}}\mathbf{x}_i+b)y_i-1\right)=0$$

Dual for separable case

Dual formulation

$$\underline{\underline{\max_{\boldsymbol{\alpha}\geq 0}}}\min_{\mathbf{w},b}\left(\frac{\mathbf{w}^{\mathsf{T}}\mathbf{w}}{2}-\sum_i \alpha_i\left((\mathbf{w}^{\mathsf{T}}\mathbf{x}_i+b)y_i-1\right)\right)$$

$$\max_{\boldsymbol{\alpha}}\left(\sum_i \alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \underbrace{\mathbf{x_i}\mathbf{x_j}}_{\text{INNER PROD.}}\right)$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

Optimality conditions
(KKT conditions)

$$\left.\begin{aligned}&\mathbf{w}=\sum_i \alpha_i \mathbf{x}_i y_i\\ &\sum_i \alpha_i y_i = 0\end{aligned}\right\}\ \text{OPTIMALITY OF } w, b$$

$$\alpha_i \geq 0$$

$$\alpha_i\left((\mathbf{w}^{\mathsf{T}}\mathbf{x}_i+b)y_i-1\right)=0$$

Dual for separable case

Dual formulation

$$\max_{\alpha}\left(\sum_i \alpha_i - \overbrace{\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j \underbrace{y_i y_j \mathbf{x_i x_j}}_{r_{ij}}}^{\frac{1}{2}\alpha^T R \alpha}\right)$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

\# VARIABLES = # data pts

\# CONSTR = # data pts + 1

NEW POINT: x

$y = \mathrm{sign}(w^T x + b)$

Optimality conditions
(KKT conditions)

$$\mathbf{w} = \sum_i \alpha_i \mathbf{x}_i y_i$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0$$

$$\alpha_i\left((\mathbf{w}^T\mathbf{x}_i + b)y_i - 1\right) = 0 \qquad / \cdot y_i$$

SOME $\alpha_i > 0$

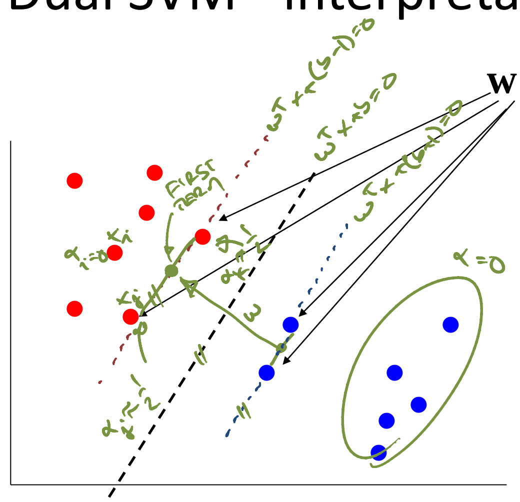
$w^T x_i + b - y_i = 0$

$b = y_i - w^T x_i$

Dual SVM - interpretation

FIRST TERM 2ND TERM

$$\left(\sum_{i\ posit}\alpha_i x_i\right) - \left(\sum_{i\ negat}\alpha_i x_i\right)$$



$$\mathbf{w} = \sum_i \alpha_i \mathbf{x}_i y_i$$

$\alpha_i > 0$

For α's that are not 0

$\in \{+1, -1\}$

$$\sum_i \alpha_i y_i = 0; \quad \alpha_i \geq 0$$

$$\sum_{i\ posit} \alpha_i = \sum_{i\ negat} \alpha_i$$

Dual SVM for linearly separable case

Our dual target function:

$$\max_\alpha \sum_i \alpha_i - \frac{1}{2}\sum_{\mathrm{i,j}} \alpha_i \alpha_j \mathrm{y}_\mathrm{i} y_j \mathbf{x_i x_j}$$

r_{ij}

ALL WE NEED ARE DOT PRODS

$$\sum_i \alpha_i \mathrm{y}_\mathrm{i} = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

Dot product across all pairs of training samples

Dot product with all training samples

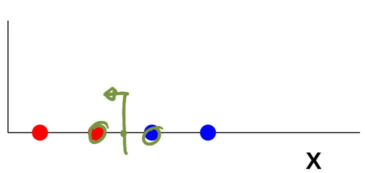
To evaluate a new sample **x** we need to compute:

$$\mathbf{w}^\mathrm{T}\mathbf{x} + b = \boxed{\sum_i \alpha_i y_i \mathbf{x}_i \mathbf{x} + b}$$

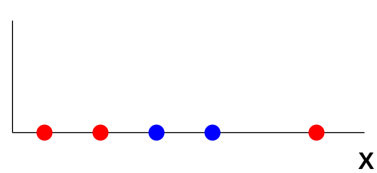
This might be too much work!
(e.g. when lifting **x** into high dimensions)

Classifying in 1-d

Can an SVM correctly classify this data?



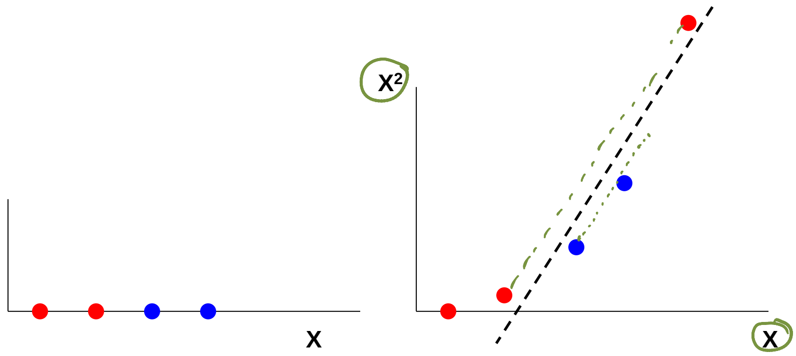
What about this?



Classifying in 1-d

Can an SVM correctly classify this data?

And now?

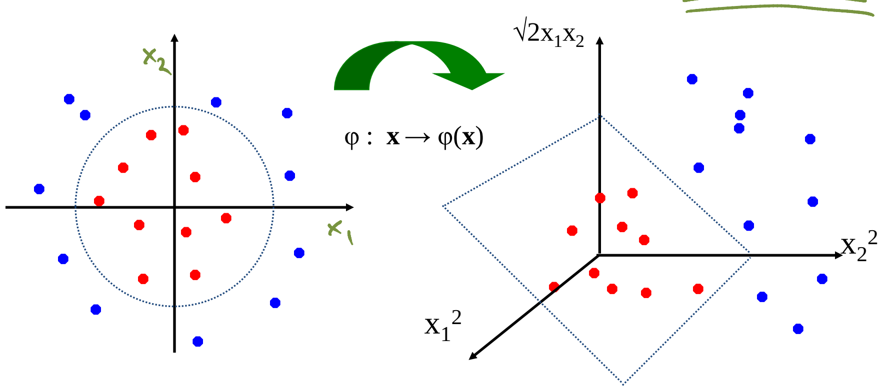


Non-linear SVDs in 2-d

- The original input space ($\mathbf{x}$) can be mapped to some higher-dimensional feature space ($\varphi(\mathbf{x})$)where the training set is separable:

$\mathbf{x}=(x_1,x_2)$

$\varphi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$

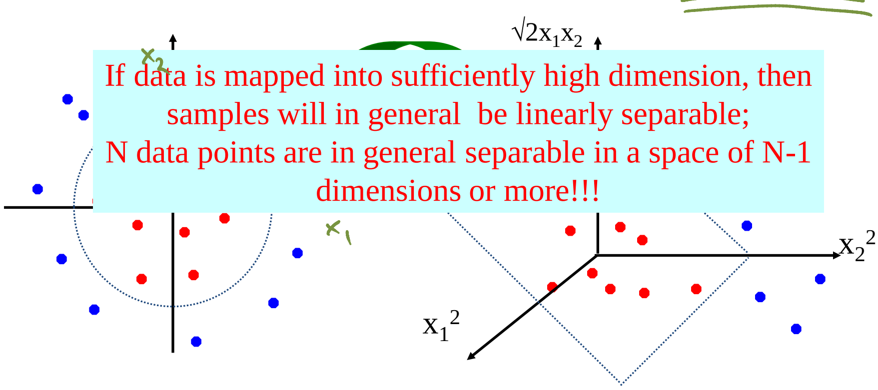


This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Non-linear SVDs in 2-d

- The original input space (**x**) can be mapped to some higher-dimensional feature space (φ(**x**))where the training set is separable:

$\mathbf{x}=(x_1,x_2)$ $\varphi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$



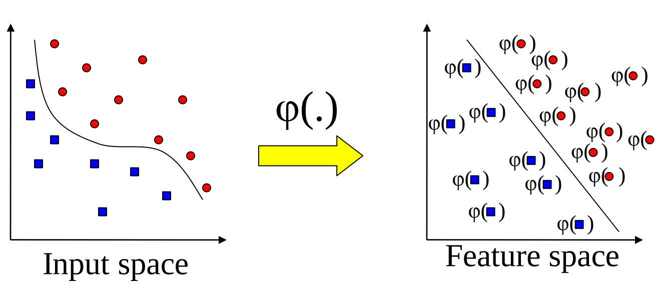
If data is mapped into sufficiently high dimension, then samples will in general be linearly separable;
N data points are in general separable in a space of N-1 dimensions or more!!!

This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Transformation of Inputs

- Possible problems
  - High computation burden due to high-dimensionality
  - Many more parameters → POSSIBLY OVERFITTING
- SVM solves these two issues simultaneously
  - "Kernel tricks" for efficient computation
  - Dual formulation only assigns parameters to samples, not features

MARGINS: solve stat. problem



Polynomials of degree two

- While working in higher dimensions is beneficial, it also increases our running time because of the dot product computation
- However, there is a neat trick we can use
- consider all quadratic terms for $x_1, x_2 \ldots x_m$

$$\max_\alpha \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \varphi(\mathbf{x_i})\varphi(\mathbf{x_j})$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

m is the number of features in each vector

The √2 term will become clear in the next slide

$$\varphi(x) = \begin{array}{c} 1 \\ \sqrt{2}x_1 \\ \vdots \\ \sqrt{2}x_m \\ x_1^2 \\ \vdots \\ x_m^2 \\ \sqrt{2}x_1x_2 \\ \vdots \\ \sqrt{2}x_{m-1}x_m \end{array}$$

$\varphi(x)$

m+1 linear terms

m quadratic terms

m(m-1)/2 pairwise terms $\approx m^2$

Dot product for polynomials of degree two

How many operations do we need for the dot product?

$$\varphi(x)\varphi(z) = \begin{array}{c} 1 \\ \sqrt{2}x_1 \\ \vdots \\ \sqrt{2}x_m \\ x_1^2 \\ \vdots \\ x_m^2 \\ \sqrt{2}x_1x_2 \\ \vdots \\ \sqrt{2}x_{m-1}x_m \end{array} \bullet \begin{array}{c} 1 \\ \sqrt{2}z_1 \\ \vdots \\ \sqrt{2}z_m \\ z_1^2 \\ \vdots \\ z_m^2 \\ \sqrt{2}z_1z_2 \\ \vdots \\ \sqrt{2}z_{m-1}z_m \end{array} = \sum_i 2x_iz_i + \sum_i x_i^2 z_i^2 + \sum_i \sum_{j=i+1} 2x_ix_jz_iz_j + 1$$

m m m(m-1)/2 **=~ m²**

Polynomials of degree d in m variables

$x_1, \ldots, x_m$

$$\varphi(x) = \begin{cases} 1 \\ x_1 \\ \vdots \\ x_m \\ \text{quad. terms} \approx m^2 \\ \text{cubic terms} \approx m^3 \\ \vdots \\ d\text{-th degree} \approx m^d \end{cases}$$

m^d

Polynomials of degree d in m variables, n data points

new dimensionality $O(m^d)$

Original formulation

Min $(w^T w)/2$

$(w^T \varphi(x_i)+b)y_i \geq 1 \quad \forall i$

dimensionality of w is $O(m^d)$

$\#$ constr $= n$

Dual formulation

r_{ij}

$$\max_\alpha \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \boldsymbol{\varphi}(\mathbf{x_i})\boldsymbol{\varphi}(\mathbf{x_j})$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

dimension of α is n

$\#$ constr $= n+1$

The kernel trick

= DOT PRODUCT

How many operations do we need for the dot product?

$$\underbrace{\varphi(x)\varphi(z)}_{K(x,z)} = \sum_i 2x_i z_i + \sum_i x_i^2 z_i^2 + \sum_i \sum_{j=i+1} 2x_i x_j z_i z_j + 1$$

m   m   m(m-1)/2   =~ m^2

$= \sim m^d$

There's **structure** to this dot product… we can do this faster!

$$\begin{aligned}(x.z+1)^2 &= (x.z)^2 + 2(x.z) + 1 \\ &= (\sum_i x_i z_i)^2 + \sum_i 2x_i z_i + 1 \\ &= \sum_i 2x_i z_i + \sum_i x_i^2 z_i^2 + \sum_i \sum_{j=i+1} 2x_i x_j z_i z_j + 1\end{aligned}$$

We only need m operations!

$(x.z+1)^d$

$= (x.z)^d + c_{d-1}(x.z)^{d-1} + c_{d-2}(x.z)^{d-2} + \vdots + c_1(x.z) + 1$

$\varphi(x) = \begin{cases} 1 \\ \sqrt{c_1}\cdot x \\ \sqrt{c_2}\cdot x^2 \\ \vdots \end{cases}$

The kernel trick

= DOT PRODUCT

How many operations do we need for the dot product?

$$\underbrace{\varphi(x)\varphi(z)}_{K(x,z)} = \sum_i 2x_i z_i + \sum_i x_i^2 z_i^2 + \sum_i \sum_{j=i+1} 2x_i x_j z_i z_j + 1$$

m   m   m(m-1)/2   =~ m^2

There's **structure** to this dot product… we can do this faster!

$$\begin{aligned}(x.z+1)^2 &= (x.z)^2 + 2(x.z) + 1 \\ &= (\sum_i x_i z_i)^2 + \sum_i 2x_i z_i + 1 \\ &= \sum_i 2x_i z_i + \sum_i x_i^2 z_i^2 + \sum_i \sum_{j=i+1} 2x_i x_j z_i z_j + 1\end{aligned}$$

We only need m operations!

Note that to evaluate a new sample we are also using dot products so we save there as well

Where we are

[lifting into poly's of degree d]

Our dual target function:

$$\max_{\alpha_i} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \underbrace{\varphi(\mathbf{x}_i)\varphi(\mathbf{x}_j)}_{O(n) \text{ operations per } (i,j)}$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

*mn*² operations to evaluate all coefficients

To evaluate a new sample **x** we need to compute:

$$\mathbf{w}^T \varphi(\mathbf{x}) + b = \sum_i \alpha_i y_i \underbrace{\varphi(\mathbf{x}_i)\varphi(\mathbf{x})}_{K(x_i, x)} + b$$

mr operations where *r* is the number of support vectors (α_i>0)

Other kernels

•Beyond polynomials there are other very high dimensional basis functions that can be made practical by finding the right k ernel function

- Radial-Basis Function:

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

- kernel functions for discrete objects (graphs, strings, etc.)

Kernels are measures of similarity

[CAVEATS]
KERNELS MUST DEFINE DOT PRODUCTS

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

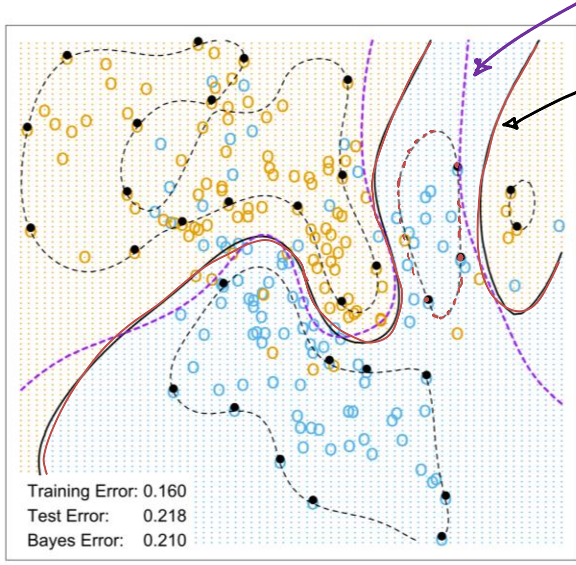
- LARGEST IF $x = z$
- DROPS AS $\|x - z\|$ INCREASES

Decision rule for a new sample **x**:

$\varphi(x_i)\cdot\varphi(x)$

$$\mathbf{w}^T\,\varphi(\mathbf{x}) + b = \underbrace{\sum_i \alpha_i y_i K(\mathbf{x}_i,\mathbf{x}) + b}_{\text{WE CARE ABOUT THE SIGN}}$$

SVM - Radial Kernel



OPTIMAL DEC. BOUNDARY

SVM DEC. BOUNDARY

$K(x,z) = \exp(\cdots)$

This slide is courtesy of Hastie-Tibshirani-Friedman, 2nd ed.

Dual formulation for non-separable case

Dual target function:

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x_i x_j}$$

$$\sum_i \alpha_i y_i = 0$$

$$C > \alpha_i \geq 0 \qquad \forall i$$

BEFORE: $C \to \infty$

The only difference is that the α_I's are now bounded

To evaluate a new sample **x** we need to compute:

$$\mathbf{w}^T \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x_i x} + b$$

Why do SVMs work?

• If we are using huge features spaces (with kernels) how come we are not overfitting the data?

- We maximize margin!

SIMPLE SOLUTIONS

- We minimize loss + regularization

$C\left(\sum_i \text{hinge loss}\right) + \text{reg.}$

TRAIN ERR

Software

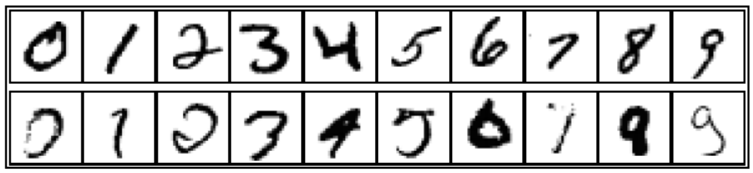
- A list of SVM implementation can be found at http://www.kernel-machines.org/software.html
- Some implementation (such as LIBSVM) can handle multi-class classification
- SVMLight is among one of the earliest implementation of SVM
- Several Matlab toolboxes for SVM are also available

Applications of SVMs

- Bioinformatics
- Machine Vision
- Text Categorization
- Ranking (e.g., Google searches)
- Handwritten Character Recognition
- Time series analysis

→Lots of very successful applications!!!

Handwritten digit recognition



3-nearest-neighbor = 2.4% error
400–300–10 unit MLP = 1.6% error
LeNet: 768–192–30–10 unit MLP = 0.9% error

Current best (kernel machines, vision algorithms) ≈ 0.6% error

Important points

- Difference between regression classifiers and SVMs'
- Maximum margin principle
- Target function for SVMs
- Linearly separable and non separable cases
- Dual formulation of SVMs
- Kernel trick and computational complexity