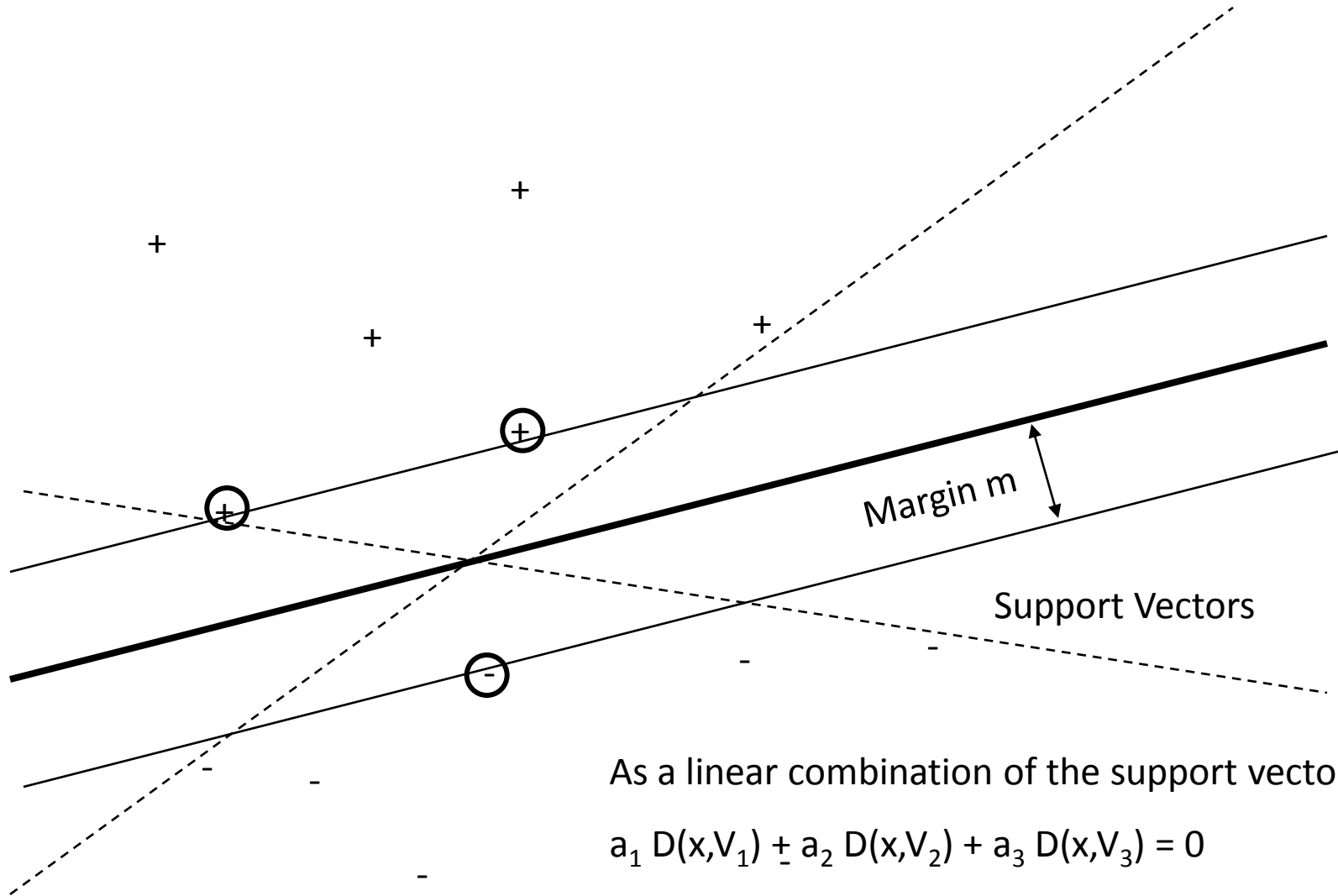


- Soon: Reinforcement Learning
- Chapters 17 & 21

What's the Best Separating Hyperplane?



As a linear combination of the support vectors

$$a_1 D(x, V_1) + a_2 D(x, V_2) + a_3 D(x, V_3) = 0$$

$$\sum a_i D(x, V_i) = 0$$

SVM

- Support Vector Machine
- Example of a kernel method
- SVM w/ Gaussian kernel is often a good generic choice for a classifier

(Although it's usually better to understand your problem and use a specialized technique)

Consider a Perceptron for Handwritten Digit Recognition

0	0	0	0	0	0
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	4	4	4	4	4
5	5	5	5	5	5
6	6	6	6	6	6
7	7	7	7	7	7
8	8	8	8	8	8
9	9	9	9	9	9

- Pixel input, e.g.:
- 32 x 32 x 8
- x = vector of feature values
- 1024 features / dimensions, each 256 values
- Generic ANNs work poorly
- Specially designed ANNs work very well
- Multi-class from binary
 - Ten index classifiers
 - All pairs w/ voting
 - Four base 2 encoders
 - Consider “3” vs. “6”
- Will a perceptron work well? Why?

Kernel Spaces: Better Metrics

- Instead of adding perceptron layers, choose a better metric (uses similarity rather than distance)
- What???
- Want 7's to be close, 8's to be close but far from 2's, etc.
- Image distance as combination of independent pixel distances does not work well
- What are we missing with perceptrons?
 - Pixels do not contribute independently
 - Must appreciate interactions among pixels

Kernel Methods

- Map to a new higher dimensional space
 - Introduces many more (interaction) features
 - Dimensionality can be very high
 - Can be infinite
- Kernel functions
 - Introduce high dimensionality
 - Complexity is independent of the inflated dimensionality
 - Defined w/ dot product of input feature vectors
(information on the Cosine between images)
- A kernel function defines a similarity metric over space of examples

Mercer's Condition / Representer Theorem

- The hypothesis space is represented efficiently by using some of the training examples – the support vectors
- The desired hyperplane is determined by

$$\sum_{i=1}^m \alpha_i K(\mathbf{s}_i, \mathbf{x}) + b$$

Dual

$$\sum_{i=0}^n w_i x_i$$

Primal

- Dual
 - Linear weighted sum of similarities to support vectors
 - Not all primal hyperplanes have a dual form
 - But the maximum margin one for this K *does*

Common Kernels

Kernel matrix positive semidefinite: dual is solution to a convex optimization problem (effic. iter. methods)

Usually start with a kernel rather than features

$(s \cdot x)$ Linear or string

$(s \cdot x)^d$ Homogeneous polynomials

$(s \cdot x + 1)^d$ Complete polynomials

$\text{Exp}(-||s - x||^2 / 2 \sigma^2)$ Gaussian / RBF

c is a constant, if K & k are kernels, so are

$K + c$ $c \cdot K$

$K + k$ $K \cdot k$

Cubic Polynomial Kernel

- $K(x,y) = (x \cdot y)^3$ or $(x \cdot y + 1)^3$
- Dot product \rightarrow scalar; [add 1]; cube it
Consider how this works...

- New Features = monomials
- Sum measures correlation among all collections of three pixels (consider non-homogeneous case)

$$\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + x_2 y_2 + x_3 y_3 + \dots + x_n y_n$$

$$\begin{aligned} (\mathbf{x} \cdot \mathbf{y})^3 &= (x_1 y_1 + x_2 y_2 + x_3 y_3 + \dots + x_n y_n) \cdot \\ &\quad (x_1 y_1 + x_2 y_2 + x_3 y_3 + \dots + x_n y_n) \cdot \\ &\quad (x_1 y_1 + x_2 y_2 + x_3 y_3 + \dots + x_n y_n) \\ &= (x_1 y_1)^3 + (x_1 y_1)^2 x_2 y_2 + \dots + x_1 y_1 x_2 y_2 x_3 y_3 + \dots + (x_n y_n)^3 \end{aligned}$$

SVMs for Digit Images

- Before 32^2 features (or about 10^3) dimensions
- Independence assumption is too strong
- Now $\sim (32^2)^3$ features (or about 10^9) dimensions
- Independence assumption is more tenable
- Linear?
 - Yes, in this high-dimensional inflated space
 - Non-linear in the original space
- Compact
 - Represented with a Kernel function and support vectors

SVMs for Digit Images

- Cubing the scalar $\sim (32^2)^3$ features
- Gaussian Kernel lives in an infinite dimensional space...
- VC(lin sep) \sim # dimensions
- Overfitting problem?
 - Not if the margin is large
 - Monitor number of support vectors

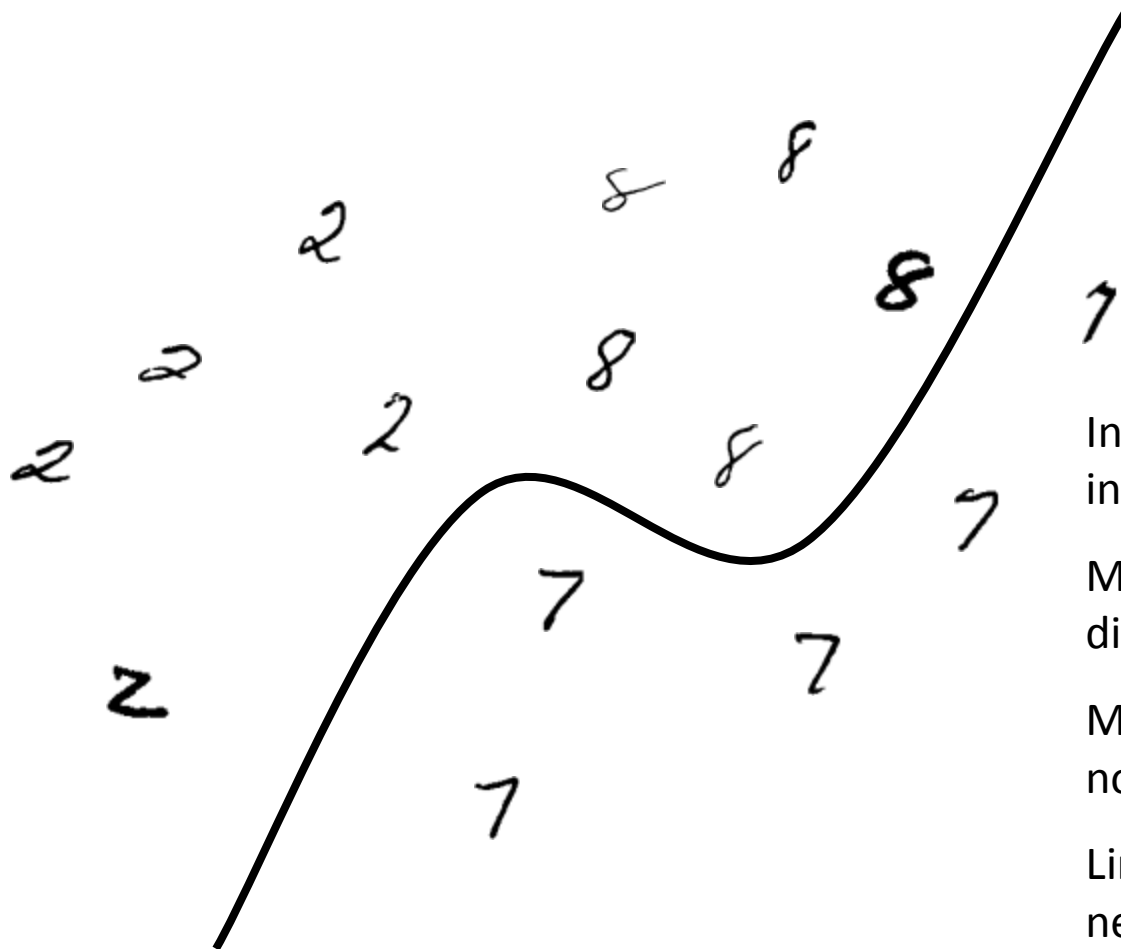
Distinguishing Handwritten Seven's vs. Two's and Eight's

Handwritten 32 x 32 gray scale pixels

Two's

Eight's

Seven's



Input feature space is inappropriate

Map inputs to a high-dimensional space

Many more features; nonlinear combinations

Linearly separable in the new space

Problems

SVMs & statistical learning generally

- Little information from each training example
 - Signal must show through the noise
 - Need many training examples
 - Thousands of are needed for handwritten digits
- Much information is ignored (weak bias vocabulary)
- Compare w/ humans
 - Novel simple shape of similar complexity
 - Master with several tens (perhaps a hundred) training examples
 - Exceedingly small non-fatigue error rate

Two Related Classification Problems



	<u>No. examples</u>	<u>error</u>	<u>No. examples</u>	<u>error</u>
Humans	< 100 ?	negligible	NA	50%
SVMs	60000	1.2%	60000	1.2%

To an SVM these are the *same problem*
Apparently the SVM ignores information crucial to people